



IVR Interface Option 8.1

IVR Server

System Administrator's Guide

The information contained herein is proprietary and confidential and cannot be disclosed or duplicated without the prior written consent of Genesys Telecommunications Laboratories, Inc.

Copyright © 1997–2012 Genesys Telecommunications Laboratories, Inc. All rights reserved.

About Genesys

Genesys is the world's leading provider of customer service and contact center software - with more than 4,000 customers in 80 countries. Drawing on its more than 20 years of customer service innovation and experience, Genesys is uniquely positioned to help companies bring their people, insights and customer channels together to effectively drive today's customer conversation. Genesys software directs more than 100 million interactions every day, maximizing the value of customer engagement and differentiating the experience by driving personalization and multi-channel customer service - and extending customer service across the enterprise to optimize processes and the performance of customer-facing employees. Go to www.genesyslab.com for more information.

Each product has its own documentation for online viewing at the Genesys Technical Support website or on the Documentation Library DVD, which is available from Genesys upon request. For more information, contact your sales representative.

Notice

Although reasonable effort is made to ensure that the information in this document is complete and accurate at the time of release, Genesys Telecommunications Laboratories, Inc., cannot assume responsibility for any existing errors. Changes and/or corrections to the information contained in this document may be incorporated in future versions.

Your Responsibility for Your System's Security

You are responsible for the security of your system. Product administration to prevent unauthorized use is your responsibility. Your system administrator should read all documents provided with this product to fully understand the features available that reduce your risk of incurring charges for unlicensed use of Genesys products.

Trademarks

Genesys, the Genesys logo, and T-Server are registered trademarks of Genesys Telecommunications Laboratories, Inc. All other company names and logos may be trademarks or registered trademarks of their respective holders. © 2012 Genesys Telecommunications Laboratories, Inc. All rights reserved.

The Crystal monospace font is used by permission of Software Renovation Corporation, www.SoftwareRenovation.com.

Technical Support from VARs

If you have purchased support from a value-added reseller (VAR), please contact the VAR for technical support.

Technical Support from Genesys

If you have purchased support directly from Genesys, please contact Genesys Technical Support at the regional numbers provided on [page 14](#). For complete contact information and procedures, refer to the [Genesys Technical Support Guide](#).

Ordering and Licensing Information

Complete information on ordering and licensing Genesys products can be found in the [Genesys Licensing Guide](#).

Released by

Genesys Telecommunications Laboratories, Inc. www.genesyslab.com

Document Version: 81iv_ad_ivr-svr_08-2012_v8.1.001.00



Table of Contents

List of Procedures	11
Preface	13
	About IVR Server.....	13
	Intended Audience.....	14
	Making Comments on This Document	14
	Contacting Genesys Technical Support.....	14
	Document Change History	15
Chapter 1	IVR Interface Option Overview	17
	IVR Interface Option 8.1	17
	New in Release 8.1	18
	Architecture	19
	IVR Server	19
	IVR Driver	23
	IVR Library.....	23
	High Availability	23
	Load Balancing	23
	Managing Service Availability.....	27
	Link Status for IVR Server	27
	DN Availability for IVR-In-Front Mode.....	27
	Agent State Management	28
	Flow Control.....	30
	Deployment Overview	30
Chapter 2	T-Server Fundamentals.....	33
	Learning About T-Server	33
	Framework and Media Layer Architecture	34
	T-Server Requests and Events	35
	Advanced Disconnect Detection Protocol	39
	Redundant T-Servers	40
	Multi-Site Support.....	44

	Agent Reservation	44
	Client Connections	45
Chapter 3	Multi-Site Support.....	47
	Multi-Site Fundamentals	48
	ISCC Call Data Transfer Service	49
	ISCC Call Flows.....	50
	ISCC Transaction Types	56
	T-Server Transaction Type Support	64
	Transfer Connect Service Feature	68
	ISCC/Call Overflow Feature	69
	Number Translation Feature.....	73
	Number Translation Rules	74
	Network Attended Transfer/Conference Feature.....	81
	Event Propagation Feature.....	83
	User Data Propagation	84
	Party Events Propagation	85
	Switch Partitioning	86
	Event Propagation Configuration	87
	ISCC Transaction Monitoring Feature	90
	Configuring Multi-Site Support.....	90
	Applications	91
	Switches and Access Codes	92
	DNs	98
	Configuration Examples.....	103
Chapter 4	Pre-Installation Setup.....	105
	IVR Interface Option Application Templates	106
	Setting up the IVR_Driver Application	106
	Configuring the IVR_Driver Application	108
	Setting up the I-Server Application	110
	Configuring the I-Server application	111
	Setting up the TServer_IVR Application	114
	Setting up the TServer_IVR_Network Application	119
	Configuring the TServer_IVR_Network Application	121
Chapter 5	Installing IVR Server.....	127
	Installing on UNIX.....	127
	Installing on Windows	128

Chapter 6	Wizard Configuration	131
	Before Using the Wizard.....	131
	Using the Wizard in Configuration Manager.....	132
	Using the Wizard Manager	133
	Wizard Manager Usage Notes.....	134
Chapter 7	Manual Configuration.....	135
	Before You Configure	136
	Preconfiguration Notes	136
	Configuration Tasks	138
	Required Configuration Tasks	139
	Logging In.....	141
	Enabling the Annex Tab.....	141
	Configuring the Switching Office	142
	Configuring Switches.....	143
	Using Access Codes.....	143
	Configuring DNSs	145
	Configuring IVRs	146
	Configuring IVR Ports.....	147
	Creating a Single IVR Port.....	147
	Configuring Multiple IVR Ports.....	148
	Configuring the Auto-Login Feature.....	150
	Configuring the I-Server Application.....	150
	Configuring the TServer_IVR Application.....	152
	Configuring the IVR_Driver Application	156
	Configuring the TServer_IVR_Network Application.....	158
	Configuring Agent Logout for Load-Sharing IVR Servers.....	162
	Configuring Peer Support	162
	Adding Servers	163
Chapter 8	High-Availability Deployment.....	165
	Warm Standby Redundancy Type	165
	Hot Standby Redundancy Type	167
	Prerequisites.....	169
	Requirements.....	169
	Synchronization Between IVR Server Pairs	169
	Warm Standby Deployment.....	170
	General Order of Deployment.....	170
	Manual Modification of Two Independent IVR Servers to a Warm Standby Pair	171
	Warm Standby Installation of IVR Server Pairs	172

	Hot Standby Deployment.....	173
	General Order of Deployment.....	173
	Manual Modification of Two Independent IVR Servers to a Hot Standby Pair.....	173
	Hot Standby Installation of IVR Server Pairs	175
	Limitations	175
Chapter 9	Starting and Stopping IVR Server.....	177
	Prestart Information	177
	Starting IVR Server.....	178
	UNIX	178
	Windows	178
	Stopping IVR Server.....	179
	UNIX	179
	Windows	179
	Starting and Stopping with Windows Services Manager	179
Chapter 10	T-Server Common Configuration Options	181
	Setting Configuration Options.....	181
	Mandatory Options	182
	TServer Section	182
	license Section	187
	agent-reservation Section.....	190
	extrouter Section	191
	ISCC Transaction Options	193
	Transfer Connect Service Options.....	197
	ISCC/COF Options	198
	Event Propagation Options	200
	Number Translation Option.....	201
	GVP Integration Option.....	202
	backup-sync Section	202
	call-cleanup Section	204
	Translation Rules Section.....	205
	security Section	206
	Timeout Value Format	206
	Changes from Release 8.0 to 8.1	207
Chapter 11	Common Configuration Options.....	209
	Setting Configuration Options.....	209
	Mandatory Options	210
	log Section.....	210

Log Output Options.....	216
Examples	220
Debug Log Options	221
log-extended Section	224
log-filter Section	226
log-filter-data Section.....	226
security Section	227
sml Section	227
common Section.....	229
Changes from 8.0 to 8.1	229
 Chapter 12	
IVR Configuration Options	231
TServer_IVR Options	231
Common Options	232
CallIdSap	233
gli	233
gli_server	234
gli_server_group_<n>	234
IServer	236
IServerGLMSap	238
pgf.....	239
pgf-debug.....	239
Timers	239
TServerClientSap	242
XmlSap	243
I-Server Options	244
AgentLogin.....	244
AgentLogout	244
InFront	245
LoadBalance	245
Stat:<stat name>	246
VirtualRoutePoints	247
IVR_Driver Options.....	248
ivr_server_interface	248
log_content	251
IVR Driver Annex Options.....	253
IVR Annex Options	255
AgentControl	255
DataTransport	255
IVR Port Annex Options	259
AutoLogin.....	259
Changes from Release 8.0 to 8.1	260

Appendix A	Sample Configurations	263
	IVR-In-Front Configuration	263
	Configuring the Switching Office.....	265
	Configuring the Switch.....	265
	Configuring the IVR	266
	Creating and Configuring the I-Server Application	267
	Configuring the TServer_IVR Application	268
	Associating the IVR with the I-Server Application.....	269
	Connecting the TServer_IVR and Stat Server Applications to the I-Server Application	269
	IVR-Behind-Switch Configuration	270
	Configuring the Switching Office.....	271
	Configuring the Switch.....	271
	Configuring the IVR	271
	Configuring the I-Server Application	273
	Configuring the TServer_IVR Application	274
	Associating the IVR with the I-Server Application.....	275
	Connecting the I-Server Application to Other Applications	275
	IVR Network T-Server Configuration	275
	Configuring the Switching Office.....	277
	Configuring the Switch.....	277
	Configuring the TServer_IVR_Network Application	278
	Connecting the TServer_IVR_Network Application to Other Applications	279
Appendix B	GLI Layer Configuration	281
	Introduction.....	281
	Concepts	281
	TCP/IP Connection	282
	GLI Protocol.....	282
	Link/Circuit Behavior.....	283
	Circuit Groups.....	283
	Circuit Failover.....	284
	Security.....	284
	Configuration	284
Appendix C	Configuring Application Connections.....	287
	IVR-In-Front Connections.....	287
	IVR-Behind-Switch Connections	288
	IVR Network T-Server Connections.....	289

Supplements	Related Documentation Resources	291
	Document Conventions	293
Index	295



List of Procedures

Implementing IVR Load Balancing	25
Activating Transfer Connect Service	69
Configuring Number Translation.	81
Activating Event Propagation: basic configuration	88
Modifying Event Propagation: advanced configuration	88
Configuring T-Server Applications	91
Configuring Default Access Codes.	93
Configuring Access Codes	94
Configuring access resources for the route transaction type	98
Configuring access resources for the dnis-pool transaction type	100
Configuring access resources for direct-* transaction types	100
Configuring access resources for ISCC/COF.	101
Configuring access resources for non-unique ANI.	101
Modifying DNs for isolated switch partitioning	102
Importing the IVR_Driver application template.	107
Defining the IVR_Driver application	107
Assigning an IVR_Driver host.	108
Defining start parameters for the IVR_Driver application	109
Adding connections for the IVR_Driver application	109
Importing the I-Server application template	110
Defining the I-Server application	111
Assigning an I-Server host	112
Defining start parameters for the I-Server application	112
Adding Connections for the I-Server Application	113
Importing the TServer_IVR application template	114
Defining the TServer_IVR application	114
Creating a virtual switching office.	115
Creating a virtual switch	116
Assigning a virtual switch	116
Assigning a TServer_IVR host	117

Defining start parameters for the TServer_IVR application	118
Enabling network logging	118
Importing the TServer_IVR_Network application template.	119
Defining the TServer_IVR_Network application	120
Creating a network switching office	121
Creating a network switch	122
Assigning a network switch	122
Assigning an IVR host	122
Defining start parameters for the TServer_IVR_Network application. .	123
Enabling network logging	124
Installing a Genesys IVR application on UNIX	128
Installing a Genesys IVR application on Windows	129
Upgrading IVR Interface Option objects to release 8.1	132
Creating a range of IVR Ports	133
Configuring new IVR Interface Option 8.1 objects	133
Enabling the Annex tab	142
Configuring the Switching Office	142
Configuring Switches	143
Adding/editing access codes	143
Configuring DNSs	145
Configuring IVRs	146
Creating a single IVR port	147
Configuring multiple IVR ports	148
Configuring the Auto-Login feature.	150
Configuring the I-Server application.	151
Configuring the TServer_IVR application.	152
Setting TServer_IVR application account permissions.	155
Configuring the IVR_Driver application	156
Setting IVR_Driver application account permissions	158
Configuring the TServer_IVR_Network application	158
Setting TServer_IVR_Network application account permissions	161
Configuring Peer Support mode for IVR Server-controlled agent login/logout	162
Adding a server	163
Modifying the primary IVR Server configuration for warm standby . . .	171
Modifying the backup IVR Server configuration for warm standby. . . .	172
Modifying the primary IVR Server configuration for hot standby	174
Modifying the backup IVR configuration for hot standby	175



Preface

Welcome to the *IVR Interface Option 8.1 IVR Server System Administrator's Guide*. This document describes the IVR Server, which is the server component of IVR Interface Option 8.1 and the IVR Software Developer's Kit (SDK). It also describes the Genesys-provided functions supported in IVR Interface Option 8.1 and the IVR SDK.

This document is valid for all 8.1 releases of this product.

Note: For versions of this document created for other releases of this product, visit the Genesys Technical Support website, or request the Documentation Library DVD, which you can order by e-mail from Genesys Order Management at orderman@genesyslab.com.

This preface contains the following sections:

- [About IVR Server, page 13](#)
- [Intended Audience, page 14](#)
- [Making Comments on This Document, page 14](#)
- [Contacting Genesys Technical Support, page 14](#)
- [Document Change History, page 15](#)

For information about related resources and about the conventions that are used in this document, see the supplementary material starting on [page 291](#).

About IVR Server

Interactive Voice Response (IVR) technology has emerged as an integral part of contact centers, financial institutions, and the travel industry. IVR components provide the initial interface when a client calls a business. Using IVRs, businesses can realize significant savings and customers can conduct their business more efficiently.

The IVR Interface Option 8.1 architecture simplifies the integration of vendor-provided IVRs with the Genesys environment. Genesys IVR Interface Option 8.1 has two major components, the IVR Server and the IVR Driver. For

more information about the IVR Interface Option 8.1 components, see Chapter 1 on [page 17](#).

Intended Audience

This guide, primarily intended for contact center administrators, contact center managers, operations personnel, and IVR developers, assumes that you have a basic understanding of:

- Computer-telephony integration (CTI) concepts, processes, terminology, and applications.
- Network design and operation.
- Your own network configurations.

You should also be familiar with the Genesys Framework architecture and functions.

Making Comments on This Document

If you especially like or dislike anything about this document, feel free to e-mail your comments to Techpubs.webadmin@genesyslab.com.

You can comment on what you regard as specific errors or omissions, and on the accuracy, organization, subject matter, or completeness of this document. Please limit your comments to the scope of this document only and to the way in which the information is presented. Contact your Genesys Account Representative or Genesys Technical Support if you have suggestions about the product itself.

When you send us comments, you grant Genesys a nonexclusive right to use or distribute your comments in any way it believes appropriate, without incurring any obligation to you.

Contacting Genesys Technical Support

If you have purchased support directly from Genesys, contact Genesys Technical Support at the regional numbers below.

Note: The following contact information was correct at time of publication. For the most up-to-date contact information, see the [Contact Information](#) on the Tech Support website. Before contacting technical support, refer to the *Genesys Technical Support Guide* for complete contact information and procedures.

Genesys Technical Support Contact Information

Region	Telephone	E-Mail
North America	+888-369-5555 (toll-free) +506-674-6767	support@genesyslab.com
Latin America	+55-11-2948-5776	la-support@genesyslab.com
Europe, Middle East, and Africa	+44-(0)-127-645-7002	support@genesyslab.co.uk
Asia Pacific	+61-7-3368-6868	support@genesyslab.com.au
Japan	+81-3-6361-8950	support@genesyslab.co.jp
India	000-800-100-7136 (toll-free) +61-7-3368-6868	support@genesyslab.com.au

Document Change History

This is the first release of the *IVR Interface Option 8.1 IVR Server System Administrator's Guide*. In the future, this section will list topics that are new or that have changed significantly since the first release of this document.

See “New in Release 8.1” on [page 18](#) for information about new functionality that is added in release 8.1.



Chapter

1

IVR Interface Option Overview

This chapter provides an overview of IVR Interface Option 8.1, describes the architecture, and provides general information about the deployment process. It also provides an overview of Load Balancing and both Warm and Hot Standby, the configuration modes that IVR Server offers for achieving redundancy and increased availability, and describes the process of implementing Load Balancing.

This chapter contains the following sections:

- [IVR Interface Option 8.1, page 17](#)
- [New in Release 8.1, page 18](#)
- [Architecture, page 19](#)
- [High Availability, page 23](#)
- [Managing Service Availability, page 27](#)
- [Deployment Overview, page 30](#)

IVR Server 8 also supports Advanced Disconnect Detection Protocol (ADDP), which you can specify on both the I-Server and TServer_IVR applications in Configuration Manager. For more information, see Chapter 2 on [page 33](#) and Chapter 10 on [page 181](#).

IVR Interface Option 8.1

IVR Interface Option 8.1 is a software application that integrates vendor-provided Interactive Voice Response (IVR) software and hardware with the Genesys Framework. It consists of three components: IVR Server, IVR Driver, and IVR Library. It is installed and configured within the Genesys 8 environment, which includes the Configuration Layer, the Management Layer, the Media Layer, and other components. For more information about these other Genesys components, refer to the specific Genesys documentation.

New in Release 8.1

The following changes have been implemented in release 8.1 of IVR Server:

- IVR Server now supports retrieval of `AttributeThirdPartyDN`.
- IVR Server now supports the Unresponsive Process Detection feature. The following configuration options enable this feature:
 - “heartbeat-period” on [page 227](#)
 - “hangup-restart” on [page 228](#)

For more information, refer to the *Framework 8.1 Management Layer User’s Guide*.

- The T-Server Common Part (TSCP) of the IVR Server logs now supports limited display of sensitive information. In logs, new options enable sensitive data to be marked for post-processing by the user, such as deletion, replacement, or hiding. For more information, see the description of the `log-filter` and `log-filter-data` configuration option sections in the *Genesys 8.1 Security Deployment Guide*.

For the IVR Server sections of the logs, the new `hide-user-data` configuration option now controls the hiding of user data within the logs, as an alternative to the `log-filter` and `log-filter-data` configuration option sections. See “hide-user-data” on [page 239](#) for details.

- IVR Server now supports IPv6. For more information, refer to the *Framework 8.1 Deployment Guide*.
- IVR Server now supports vSphere 4 Hypervisor.
- IVR Server now supports Acresso FLEXNet Publisher v11.9 license manager.
- IVR Server is now supported on the following operating system platforms:
 - IBM AIX 7.1 64-bit
 - HP-UX Itanium (version 11i v3)
 - Windows Server 2008 R2
 - Red Hat Enterprise Linux 5 32-bit and 64-bit

-
- Notes:**
- To take advantage of the full functionality of IVR Server 8.1, Genesys recommends that you use Framework 8.1 and IVR Driver 8.1.
 - Configuration option changes that apply to IVR Server are described in “Changes from Release 8.0 to 8.1” on [page 260](#).
-

Architecture

As previously mentioned, IVR Interface Option 8.1 consists of the following components:

- IVR Server
- IVR Driver

The following subsections describe each of these components in turn.

IVR Server

IVR Server, the key component of Genesys IVR Interface Option 8.1, provides the following functionality:

- Tracks call flow
- Interfaces multiple drivers with multiple T-Servers
- Works with other Genesys services (such as T-Server, Stat Server, and Universal Routing Server)
- Can be used in Load Balancing, or Warm, or Hot Standby modes

Genesys provides the following configuration modes for the IVR Server:

- IVR-Behind-Switch, a basic configuration in which a T-Server that is connected to the premise switch (using computer-telephony integration [CTI] links) can monitor the call activity on IVR channels. For more information, see “IVR-Behind-Switch Configuration” on [page 20](#).
- IVR-In-Front, in which a CTI link is not involved in the call processing. For more information, see “IVR-In-Front Configuration” on [page 20](#).
- IVR Network T-Server, in which the IVR Server (an IVR T-Server running in Network mode) is a link to a user-provided Network IVR application. The routing strategy and a Genesys Network T-Server are used to route the calls to the Network IVR for processing. For more information, see “IVR Network T-Server Configuration” on [page 21](#).

IVR Server supports the predictive dialing method used by Genesys outbound components (Outbound Contact Solution and Voice Callback Solution).

Library Usage

IVR Server uses the standard T-Server Library (T-Lib) for its interactions. IVR Server supports connection to a regular T-Server, the `TServer_IVR` function of an IVR Server operating in IVR-In-Front mode, and a Network T-Server.

IVR-Behind-Switch Configuration

In the IVR-Behind-Switch configuration, an incoming call arrives at the premise switch before it goes to the vendor-provided IVR (see [Figure 1](#)). The premise switch and T-Server are at the same site.

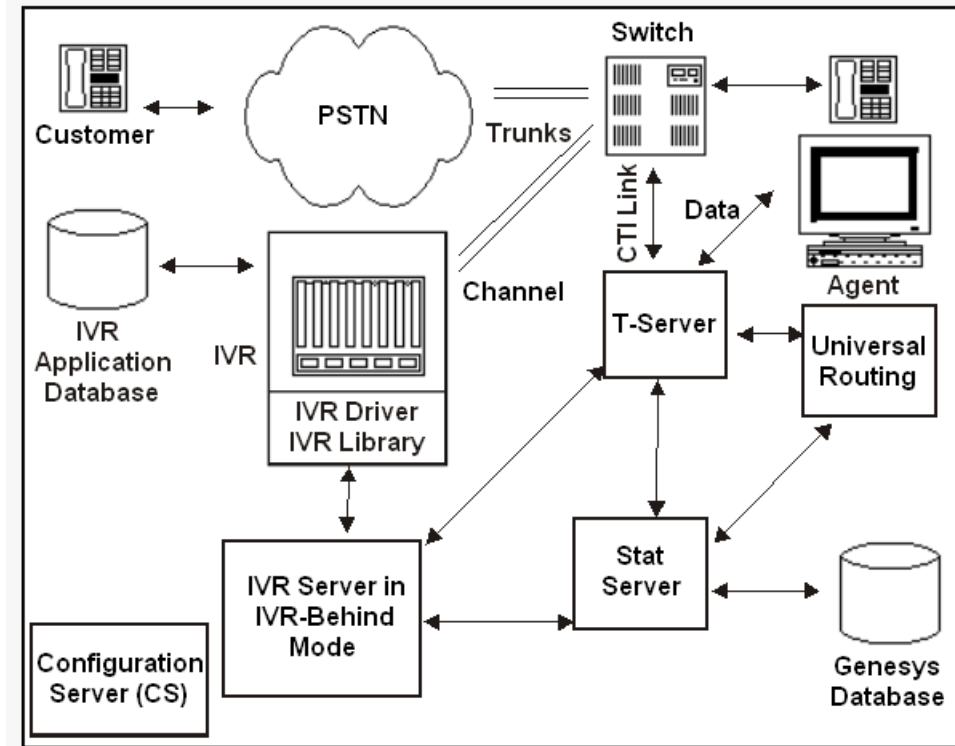


Figure 1: IVR-Behind-Switch Configuration

In this configuration, a T-Server is connected to a premise switch, and the IVR is connected directly to both the switch (through phone lines) and the IVR Server (through data lines). The IVR Server communicates with the T-Server and the Stat Server.

IVR-In-Front Configuration

When a vendor-provided IVR is connected directly to the PSTN (Public Switched Telephone Network), without a premise switch, the configuration is called IVR-In-Front. In the Site A configuration shown in [Figure 2](#), there is no T-Server to connect to, because there is no premise switch.

In the IVR-In-Front configuration, the TServer_IVR function resides within the IVR Server.

IVR Server operating in IVR-In-Front mode supports IVRs that are connected directly to a PSTN, by performing functions similar to a regular T-Server. When an IVR is considered a termination point for incoming calls, no premise

switch is involved, and no local T-Server receives notification of the incoming call. Instead, IVR Server provides this functionality.

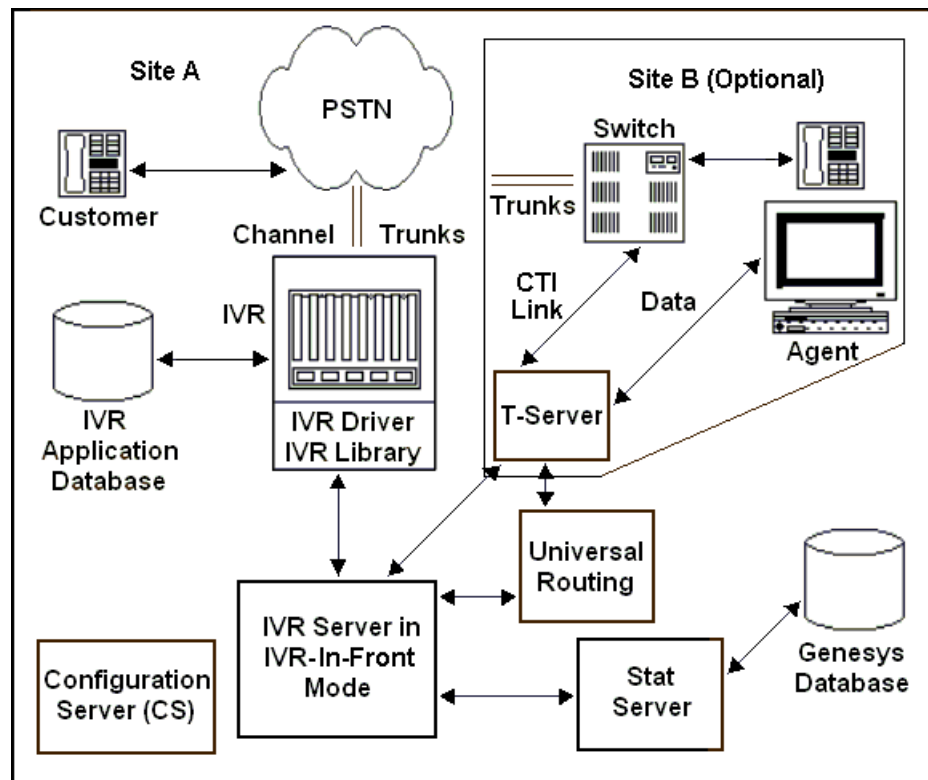


Figure 2: IVR-In-Front Configuration

In the IVR-In-Front configuration shown in [Figure 2](#), Site A is configured for IVR-In-Front mode. The IVR Server communicates with the IVR, the Universal Routing Server (URS), and the Stat Server. The IVR Server also simulates a T-Server and can communicate with other T-Servers, such as the T-Server at Site B. The IVR at Site A is physically connected to the public telephony network for phone lines, and to the IVR Server for data lines.

Site B includes a physical switch connected to a physical T-Server, which, in turn, provides data to agents in an agent pool.

This distributed configuration across Sites A and B enables coordinated Call Data Transfers.

IVR Network T-Server Configuration

When a vendor-provided IVR and a Network T-Server are connected directly to a PSTN, without a premise switch, the configuration is called IVR Network T-Server (see [Figure 3](#)).

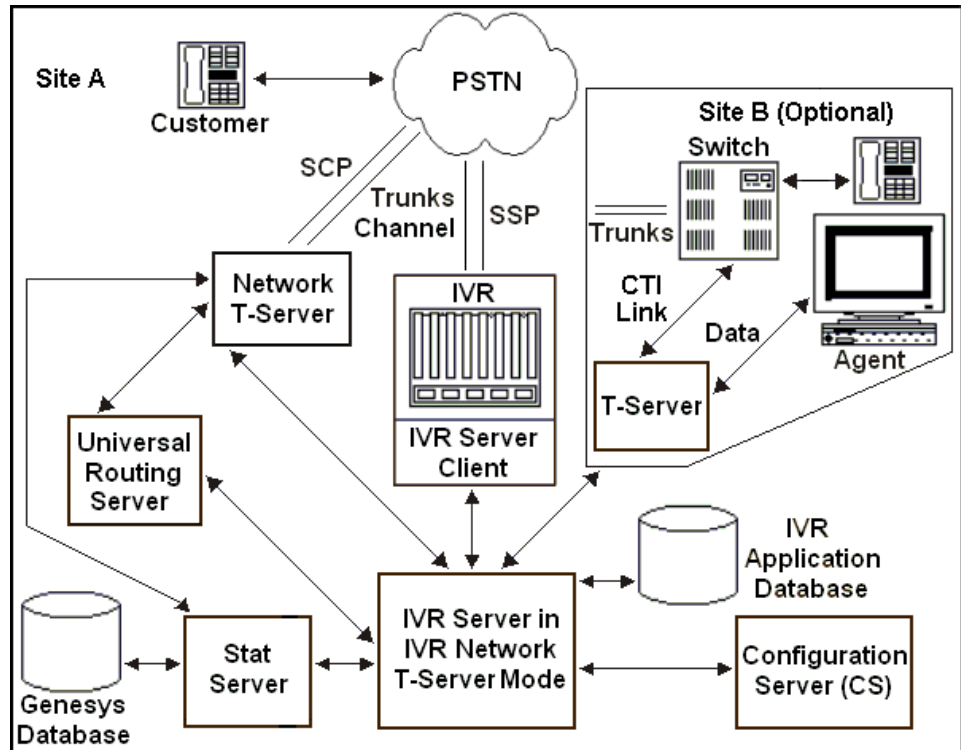


Figure 3: IVR Network T-Server Configuration

1. The PSTN sends a request for a destination to the Network T-Server, which forwards the request to URS.
2. URS responds with a destination address to the Network T-Server, which forwards the response to the PSTN.
3. The call is then routed from the PSTN to the IVR.
4. The IVR notifies the IVR Server that a new call has arrived.
5. The IVR Server then tells the IVR which treatments to apply for this call.
6. When an available agent is located at the destination address specified by URS, the call is routed to the premise T-Server where the agent's desktop resides.

Genesys IVR Drivers do not support the IVR Network T-Server configuration for IVR Server. Therefore, in this configuration mode, an IVR Server client application is used in the place of a Genesys IVR Driver for communication between the vendor-provided IVR and the IVR Server. You can use the IVR XML SDK to create your own IVR Server client application.

IVR Server (in Network mode) supports the Network Call Monitoring feature, which enables the CTI-less T-Server of the Expert Contact Solution to receive Call Monitoring Events such as call created, call deleted, call party added, and call party deleted (see the *Genesys Expert Contact Solution* documentation).

IVR Driver

The IVR Driver component integrates vendor-specific IVR hardware and software with the Genesys environment. It adds to the IVR user interface a set of functions or calls that can be used to generate scripts and to integrate the vendor-provided IVR with the Genesys environment. All interactions between the IVR Driver and other IVR Interface Option components are based on the request-response architecture of the IVR Library and use a TCP/IP connection.

The major functions provided by the IVR Driver include:

- Telephony function support (such as transfer, conference, answer, and release).
- Call data manipulation (such as attach, update, and delete).

Each vendor-provided IVR needs one Genesys IVR Driver in order to operate in the Genesys environment. If you run vendor-provided IVRs from various manufacturers, each IVR must have a corresponding IVR Driver that is designed for it.

IVR Library

The IVR Library component is embedded in Genesys IVR Drivers. Typically, it is used to return any IVR Driver error messages.

With IVR Interface Option 8.1, the IVR Library's communication interface uses the industry-standard XML (eXtensible Markup Language) protocol for the transport layer. For more information about the XML interface, see the *IVR SDK 8.1 XML Developer's Guide*, which is available only with purchase of the Genesys IVR SDK.

For more information about the IVR Library interface, see the *Genesys Developer Program 8.1 IVR SDK C Developer's Guide*.

High Availability

There are three different ways to achieve high availability with IVR Server:

- Warm standby
- Hot standby
- Load Balancing

Warm and hot standby are discussed in detail in Chapter 8, "High-Availability Deployment," on [page 165](#).

Load Balancing

You enable IVR Server Load Balancing by creating multiple, separate IVR Servers. You need both a TServer_IVR and an I-Server application for each

IVR Server in the Load Balancing group. One IVR Server must be designated as the primary, with all the others configured as secondary.

Note: The primary IVR Server is designated by using the `app-name` option, in the `LoadBalance` section on the `Options` tab of each *secondary* IVR Server's `Properties` dialog box. No configuration setting is required in the primary IVR Server's `Properties` dialog box.

IVR Server clients that are implemented with the Genesys IVR Library (including all the Genesys IVR Drivers from 7.5+) route calls to the IVR Servers in a Load Balancing group according to the following formula:

$$\langle \text{number of ports} \rangle \bmod \langle \text{number of active IVR Servers} \rangle$$

When one IVR Server fails, this configuration enables the surviving IVR Server(s) to continue handling their current calls, and new incoming calls are distributed according to this formula, with the number of active IVR Servers now decreased by one. When the failed IVR Server is restored, it is automatically added back into the distribution.

To create a different routing algorithm for IVR Server Load Balancing, you can write your own interface by using the Genesys IVR SDK XML. See the specific IVR Driver documentation for details.

If a software failure occurs on any IVR Server in the group (including the primary), any calls that are already in progress on that IVR Server are lost, but all the other IVR Servers continue to operate. Subsequent calls are distributed to the rest of the IVR Servers, bypassing the one that is down. When that IVR Server comes back online, calls are distributed to it again.

The IVR Servers in the Load Balancing group can reside on either the same machine or multiple machines. Because no limit is imposed on the number of IVR Servers that you can include in a Load Balancing group, this configuration method is highly scalable.

Note: In Load Balancing (but not Warm Standby), the secondary IVR Servers are not controlled by SCI switchover. Although the primary and secondary IVR Servers can be started in any order, the primary must be running before IVR Drivers 8 for Aspect Unixware and CONVERSANT can connect to any IVR Server. Other drivers can connect to IVR Servers that have been started in any order.

Implementing IVR Server Load Balancing

In general, load-balanced IVR Servers cannot be the target of an ISCC transfer. However, the following case is an exception.

Exception

Load-balanced IVR Servers deployed in In-Front mode can be designated as destination locations for Inter Server Call Control (ISCC), with the transaction type set to `dnis-pool`. The destination load-sharing TServer_IVR applications must be configured on the `Connections` tab of the source T-Server.

Note: Universal Routing Server (URS) version 7.5.002.02 or later is a required element within the solution. There is no configuration information required for IVR Server.

This configuration information must be entered in the default section on the `Options` tab of the URS Server application as follows:

- `use_extrouter = false`—URS delegates routing functionality to IVR Server.
- `use_extrouting_type = dnis`—URS is required to provide an access number that is to be used at the remote site.

For more details on this feature, refer to the *Universal Routing Server 8.1 System Administrator's Guide*.

Limitation

T-Server Common Part does not support multiple source to multiple targets, only many-to-one or one-to-many. Therefore, deployments of load-balanced IVR Servers to load-balanced T-Servers, for instance, are not supported.

Procedure: Implementing IVR Load Balancing

Start of procedure

1. For each of the secondary I-Server applications, open the `I-Server_810 Properties` dialog box, click the `Options` tab, and do the following:
 - In the `LoadBalance` section, enter the following:
`app-name <primary IVR Server application name>`
 For example:
`app-name I-Server_810`
 - If you have an IVR-Behind-Switch configuration and you are using Virtual Routing Points, create Virtual Routing Point entries. These entries can include alias names.

Example 1—Basic

For load-balancing in Behind mode only, only one IVR Server can control a Route Point at any one time, so an alias list must be used. Configure such a list as follows:

```
[VirtualRoutePoints]
```

```
tsname=vrp1:vrp2:vrp3
```

Here, `tsname` is the T-Server application on which the Virtual Routing Points are defined, and `vrp1`, `vrp2`, and `vrp3` are the Virtual Routing Point DN names.

Example 2—Alias

This syntax is still supported, but it has been extended to enable you to assign an alias name to a Virtual Routing Point. For example, to assign the alias name `sales` to `vrp1`, code the following:

```
tsname=vrp1|sales:vrp2:vrp3
```

From the IVR Driver, you can then send a `RouteGet` on DN `sales`, and this would be translated into a route request on Virtual Routing Point `vrp1`.

This feature becomes important when you are configuring Load Balanced IVR Servers, because it is not possible for two (or more) IVR Servers to use the same Virtual Routing Point. In other words, each IVR Server *owns* the Virtual Routing Points with which it is configured. In this case, you would configure two Virtual Routing Points—one for each IVR Server—and give them the same alias name. In this way, the IVR client can balance the load between multiple IVR Servers, and use the same logical Route Point for routing operations (that is, the Virtual Routing Point alias name).

For example, on IVR Server 1, code the following:

```
[VirtualRoutePoints]
    tsname=vrp1|sales
```

On IVR Server 2, code the following:

```
[VirtualRoutePoints]
    tsname=vrp2|sales
```

Route requests on the `sales` alias to IVR Server 1 would be issued on `vrp1`, and route requests on the `sales` alias on IVR Server 2 would be issued on `vrp2`.

Example 3—More Than 255 Virtual Routing Points

To configure more than 255 Virtual Routing Points for an I-Server application, segment the Virtual Routing Point definitions by using an extension delimiter:

```
[VirtualRoutePoints]
<tsname><ExtensionDelimiter><ID> = vrp1|sales...
```

In processing, `<ID>` is stripped, and the definitions to the right of the equal sign are merged with the other options that are specified in the `VirtualRoutePoints` section, on the `Options` tab of the T-Server that is specified by `<premise T-Server application name in CME>` (shown as `<tsname>` in this example). For more information, see the description of the `VirtualRoutePoints` section on [page 247](#).

2. If you are using a Genesys IVR Driver 7, you must configure Load Balancing in the `IVR_Driver` application as well. For more information, see “IVR_Driver Options” on [page 248](#).

End of procedure

Note: If you desire to run load-balanced IVR Servers in Network mode, you just configure multiple IVR Servers onto the same switch as there is no `LoadBalance` section.

Managing Service Availability

In the management of a complex contact center environment, it is occasionally necessary to remove individual IVR ports, entire IVRs, or entire IVR Server applications from service, due either to planned maintenance, or to unplanned hardware or software failures. This section describes the various methods for managing service availability for IVRs.

Link Status for IVR Server

When there are no IVR Driver clients connected to IVR Server, an `EventLinkDisconnected` message is sent to all connected T-Library clients. For IVR-In-Front and IVR Network T-Server configurations, this prevents the Genesys Framework from routing calls to this location.

For configurations in which only a single IVR object is associated with an IVR Server application, calls will be prevented from being sent to this IVR from another Genesys contact center when the IVR Driver is not connected to the IVR Server and ready to process calls.

However, for configurations in which multiple IVR objects are associated with an IVR Server application, calls can be routed to this IVR Server when any one or more IVR Drivers are connected to the IVR Server. In this case, calls can still be routed to an IVR when the IVR Driver that serves it is not operational.

For configurations in which multiple IVR objects are associated with a single IVR Server application, you must take additional steps to prevent calls from being routed to an IVR when the IVR Driver is not operational. These steps are discussed in the following subsections.

DN Availability for IVR-In-Front Mode

For IVR-In-Front configurations that are operating in Warm Standby or stand-alone mode, you can configure IVR Server to report DN status events (`EventDNOutOfService` and `EventDNBackInService`). If you set the `report-dn-status` configuration option (see [page 237](#)) to true, IVR Server will

report the DNs that are associated with an IVR as out of service when the IVR Driver is not connected to IVR Server. After the IVR Driver connects to IVR Server, the DNs that are associated with the IVR are placed back in service.

When DNs are reported as out of service, calls from other Genesys contact centers will not be routed to the IVR that serves these IVR Ports if the driver is not ready to service new calls.

Note: This feature is not supported for load-balanced IVR Server configurations.

Agent State Management

All prior releases of IVR Server have provided the capability to associate a contact center agent with an IVR Port and to manage the associated agent status. This enables advanced call routing techniques such as skills-based routing and agent availability. By default, this behavior remains unchanged since release 7.2, and existing deployments will continue to work in the same way as in previous releases (see the description of the `LegacyMode` configuration option on [page 255](#)). However, there are some features related to agent state management that were introduced starting with release 7.2. You must change the following configuration options in order to use these features:

- `report-dn-status` (see [page 237](#))
- `LegacyMode` (see [page 255](#))
- `LogoutOnDisable` (see [page 245](#))
- `flow-control` (see [page 236](#))

For configurations in which agents are associated with IVR Port objects (see the description of the `AgentLogin` section of the IVR Port configuration object's Annex tab on [page 245](#)), the default system behavior is for IVR Server to control agent state, beginning at application startup and extending to the time when IVR Server is shut down. This works well when all the IVR Drivers are available and able to form network connections to IVR Server. However, if the IVR Driver is not able to connect to IVR Server for any reason, the agents appear as logged in and ready to receive calls, but any calls that are routed to this IVR cannot be processed properly. To resolve this issue, two features were introduced starting with release 7.2: Agent Logout and Agent State Status Management.

Agent Logout

The first feature enables agent logout to be triggered when IVR Port or IVR configuration objects are disabled. This provides a convenient way to manually make specific IVR ports unavailable, or to make all IVR ports that are associated with an IVR unavailable.

This feature is available only when IVR Server is controlling agent status (that is, when the `LegacyMode` option (see [page 255](#)) is set to `true`, which is the default value), and when the `LogoutOnDisable` configuration option (see [page 245](#)) is set to `true`.

When an IVR port is disabled in this configuration, the IVR Server sends a `TAgentLogout` request, and then stops monitoring events for the DN that is assigned to this IVR port. When an IVR port is enabled, IVR Server begins monitoring the DN for this IVR port again, and places the agent into the proper state.

Note: Enabling/disabling an IVR object has the effect of enabling/disabling all the configured IVR ports for this IVR.

Agent State Status Management

The second feature transfers responsibility for agent state status management from IVR Server to the IVR Driver application.

This feature is available for select IVR Drivers. The `LegacyMode` configuration option (see [page 255](#)) of an IVR must be set to `false`. In this mode, the IVR Server application does not perform any of the following agent-related requests for DNs that are associated with this IVR:

- `login`
- `logout`
- `ready`
- `notready`

Instead, the IVR Driver application manages agent status directly. This eliminates situations in which IVR Server logs in agents before the IVR Driver is ready to receive calls.

To determine whether this feature is supported, and to find information about configuring the driver application to support it, consult the documentation for your particular driver.

The `LegacyMode` option is configured on an IVR object, and multiple IVR objects can be associated with a single IVR Server. Therefore, agent state management can be performed by IVR Server in some cases, and by IVR Driver in other cases. However, for a single IVR, agent status is controlled exclusively by either IVR Server (`LegacyMode = true`) or IVR Driver (`LegacyMode = false`).

Note: The `LogoutOnDisable` configuration option (see [page 245](#)) has no effect on agents associated with an IVR object that is configured with `LegacyMode` set to `false`.

Flow Control

To enable an IVR Server application instance to be taken out of service with no impact on existing calls, the flow control feature was introduced starting with release 7.2.

Note: In order for this feature to function properly, the IVR Server and IVR Driver applications must support it. To determine whether flow control is supported, consult the documentation for your particular driver.

Flow control enables IVR Server to instruct all connected IVR Drivers to stop delivering new calls. In a Load Balanced configuration, these calls are directed to another instance of the IVR Server application. This means that service for new calls is not affected. Calls that have already started or that are in progress when flow control is enabled proceed normally. Then, after all these calls have been handled normally, it is safe to shut down the IVR Server application. For more information, see the description of the `flow-control` configuration option on [page 236](#).

Deployment Overview

The IVR Interface Option 8.1 deployment process includes the installation and configuration of the IVR Server and IVR Driver.

Note: In order for IVR Interface Option 8.1 and/or IVR SDK to operate properly, you must first install and configure the other components in the Genesys 8 environment (the Configuration Layer, Management Layer, Media Layer, and so on). For information about how to deploy these components, see the specific Genesys documentation.

For information about how to install the IVR Driver, see the *IVR Interface Option 8.1 System Administrator's Guide* for your particular IVR Driver.

For detailed information about the IVR SDK, see the *IVR SDK 8.1 C Developer's Guide*, the *IVR SDK 8.1 XML Developer's Guide*, and the *IVR API Reference Tree* help file.

Before you install IVR Interface Option 8.1 on Windows and UNIX operating systems, verify that a supported Genesys Framework release is installed, configured, and running in your computing environment.

IVR Server 8.1 is interoperable and compatible with specific earlier releases of Genesys Framework and IVR Drivers, as well as with customer applications

built with the IVR SDK. You can use IVR Server 8.1 with releases 6.5, 7.0, 7.1, 7.2, 7.5, and 8.0 of these products.

Note: If you use IVR Server 8.1 with earlier Genesys products and custom applications, the functionality of IVR Server 8.1 will be limited to that which was available for those particular releases.

Prior to installation, verify that the computers on which the IVR Interface Option 8.1 components will operate meet the minimum hardware and software requirements. Consult the following resources:

- The *Genesys Migration Guide*, which contains a documented migration strategy for each software release. Refer to the applicable portion, or contact Genesys Technical Support for additional information.
- *Genesys Supported Media Interfaces* and *Genesys Supported Operating Environments Reference Guide*, which provide lists of the supported IVRs, operating systems, and databases.
- The *Genesys Licensing Guide*, for detailed information about the licensing that IVR Server requires. Refer to the applicable portion, or contact Genesys Technical Support for additional information.

All these documents are available on the Genesys Technical Support website at <http://genesyslab.com/support>.



Chapter

2

T-Server Fundamentals

This chapter provides general information about T-Server features and functionality and about its configuration and installation.

This chapter has various levels of information, some of it intended for people who have configured, installed, and used previous releases of T-Server, and some of it aimed at those less familiar with such T-Server operations. That means some sections will not necessarily be relevant for you.

Generally, this chapter presents overview information that applies to all T-Servers (and Network T-Servers) and their deployment. This chapter is divided into the following sections:

- [Learning About T-Server, page 33](#)
- [Advanced Disconnect Detection Protocol, page 39](#)
- [Redundant T-Servers, page 40](#)
- [Multi-Site Support, page 44](#)
- [Agent Reservation, page 44](#)
- [Client Connections, page 45](#)

Learning About T-Server

The *Framework 8.1 Deployment Guide* provides you with a high-level introduction to the role that T-Server plays in the Genesys Framework. If you have already looked through that guide, you may recall that T-Server is the most important component of the Framework Media Layer (the other two components are Load Distribution Server (LDS) and HA Proxy). The Media Layer enables Genesys solutions to communicate with various media, including traditional telephony systems, voice over IP (VoIP), e-mail, and the Web. This layer also provides the mechanism for distributing interaction-related business data, also referred to as *attached data*, within and across solutions.

Framework and Media Layer Architecture

Figure 4 illustrates the position Framework holds in a Genesys solution.

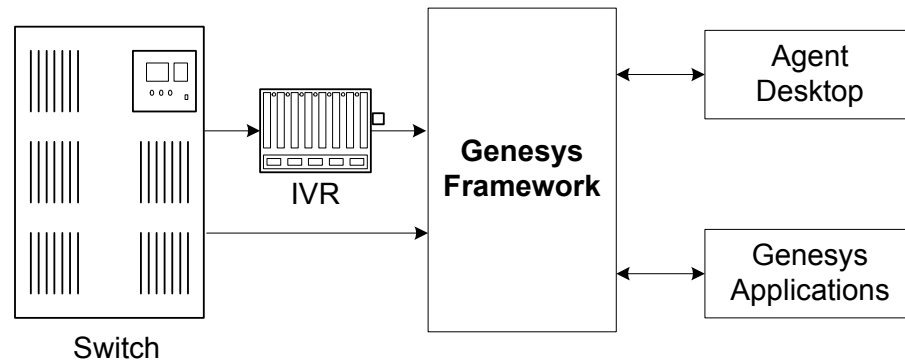


Figure 4: Framework in a Genesys Solution

Moving a bit deeper, Figure 5 presents the various layers of the Framework architecture.

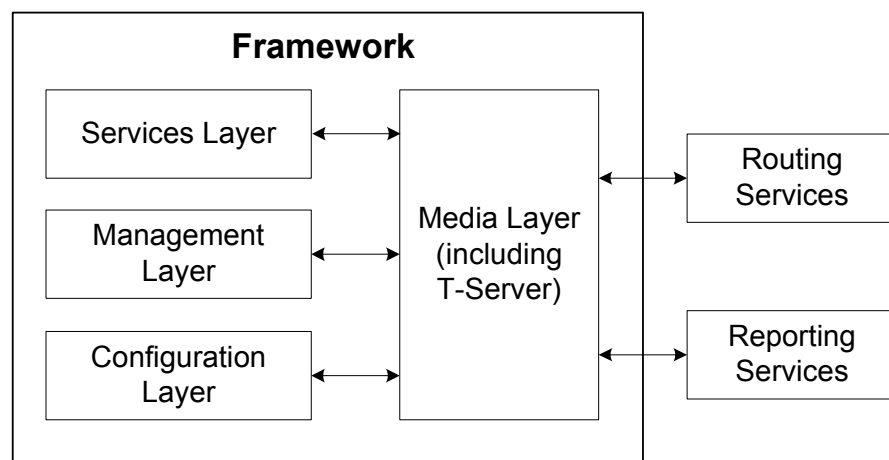


Figure 5: The Media Layer in the Framework Architecture

T-Server is the heart of the Media Layer—translating the information of the media-device realm into information that Genesys solutions can use. It enables your contact center to handle the computer-based form of the interactions that arrive and it translates the information surrounding a customer contact into reportable and actionable data.

Figure 6 presents the generalized architecture of the Media Layer.

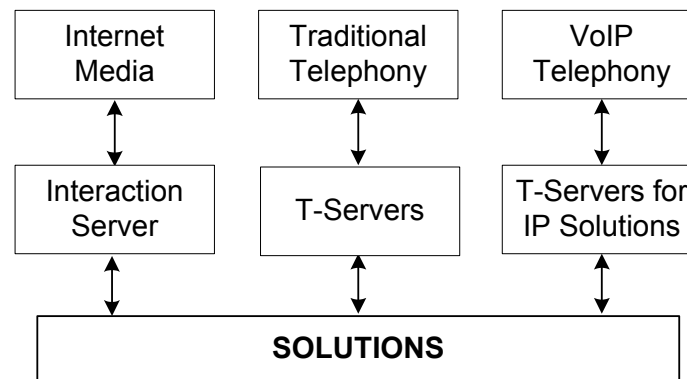


Figure 6: Media Layer Architecture

In addition to being the most important component of the Media Layer, T-Server plays the most significant role in making information about telephony traffic and its data available to Framework as a whole.

One or more components in practically every solution are T-Server clients. Solutions comprise a number of different Genesys software packages, from collections of components for various types of routing to those that allow for outbound dialing to still others. Framework in general, and T-Server in particular, enable these solutions to function in your enterprise.

T-Server has several typical clients: Stat Server, Interaction Concentrator, Universal Routing Server, and agent desktop applications. T-Server gets the information it needs about the enterprise from Configuration Server. Additionally, if you use the Management Layer, T-Server provides its ongoing status and various other log messages to server components of the Management Layer (for instance, allowing you to set alarms).

T-Server Requests and Events

This section outlines the roles that T-Server plays in a contact center. While it is possible to describe roles for all T-Servers, at a detailed level, T-Server's functionality depends on the hardware to which it is connected. (For example, when connected to a traditional switch, it performs CTI functions, but when connected to a VOIP-based telephony device, it controls IP traffic.) The CTI connection is only for the switch.

Details of T-Server Functionality

T-Server is a TCP/IP server that enables intelligent communication between media-specific protocols (such as the various CTI protocols, including CSTA and ASAI) and TCP/IP-based clients of T-Server. Applications that are clients

to T-Server use the T-Library format to transmit requests to T-Server through a TCP/IP socket. T-Server can then either translate those requests to CTI protocol for switch use or relay them directly to other TCP/IP clients.

T-Server performs three general functions in the contact center: Bridging, Messaging, and Interaction Tracking.

Bridging

T-Server acts as a platform-independent interface between media devices and business applications. In the case of a telephony device, for instance, it receives messages from and sends commands to the telephony equipment using either CTI links provided by the switch manufacturer or interface protocols provided by telephony network vendors.

On the client-application end, T-Server offers three models (call model, agent model, and device model) unified for all switches. The core functionality (such as processing an inbound call, an agent login, or a call-forwarding request) translates into a unified application programming interface (API) called T-Library, so that applications do not need to know what specific switch model they are dealing with. On the other hand, T-Library accommodates many functions that are unique to a specific switch, so that client applications are able to derive the maximum functionality offered by a particular switch.

Refer to the *Genesys Events and Models Reference Manual* for complete information on all T-Server events and call models and to the `TServer.Requests` portion of the *Voice Platform SDK 8.x .NET (or Java) API Reference* for technical details of T-Library functions.

Messaging

In addition to translating requests and events for the client application involved in an interaction, T-Server:

- Provides a subscription mechanism that applications can use to receive notifications about interaction-related and non-interaction-related events within the contact center.
- Broadcasts messages of major importance (such as a notification that the link is down) to all clients.
- Broadcasts messages originated by a T-Server client to other T-Server clients.

The subscription mechanism consists of two parts, the DN subscription and event-type masking. Applications must register for a DN or a set of DNs to receive notifications about all events that occur in association with each registered DN. For example, when two softphone applications are registered for the same DN, and the first application initiates a call from the DN, T-Server notifies both applications that the call is initiated from the DN.

Client applications can also specify one or more types of events, and T-Server will filter out events of the non-specified types and only send events of the

requested types. For example, if agent supervisors are interested in receiving agent-related events, such as AgentLogin and AgentLogout, they have to mask EventAgentLogin and EventAgentLogout, provided that a particular T-Server supports these events.

The combination of each client's subscription for DNs and masking of event types defines what messages T-Server distributes to what client.

Interaction Tracking

T-Server maintains call information for the life of the call (or other T-Server-supported media type) and enables client applications to attach user data to the call. Call information includes:

- A unique identifier, connection ID, that T-Server assigns when creating the call.
- Automatic Number Identification (ANI) and Dialed Number Identification Service (DNIS), if reported by the CTI link.
- User data that a client application (such as an Interactive Voice Response unit or Genesys Universal Routing Server) provides.

Difference and Likeness Across T-Servers

Although Figure 6 on [page 35](#) (and other figures) depicts T-Server that works with telephony systems as a single product, this is a simplification. Because almost every traditional telephony device has its own characteristics and communication protocols, Genesys makes different T-Servers for different telephony systems. (That means your T-Server will not work with another switch.) Thus, all T-Servers play a common role in the architecture, but their specific features differ from implementation to implementation, based on the media device in use.

Despite their switch-based differences, T-Servers for telephony systems are similar to one another in at least one important respect: they are all built with a certain amount of shared software code. This shared code is rolled into a single unit and is called T-Server Common Part (TSCP). TSCP is the central, common component for all T-Servers and has its own Release Note, which is accessible via a hyperlink from your T-Server's Release Note.

Note: This document separates common-code features based on TSCP into separate sections and chapters, such as the “T-Server Common Configuration Options” chapter. These are the options for all T-Servers that TSCP makes available for configuration.

T-Server Functional Steps During a Sample Call

The following example, [Figure 7](#), outlines some basic steps that T-Server might take when a call arrives from outside the contact center. In this scenario,

T-Server starts tracking the call even before it is delivered to the agent. T-Server then informs the selected agent that a call has arrived. When the switch delivers the call to the agent's extension, T-Server presents account information, collected at an Interactive Voice Response (IVR) unit, to the agent at the agent desktop application.

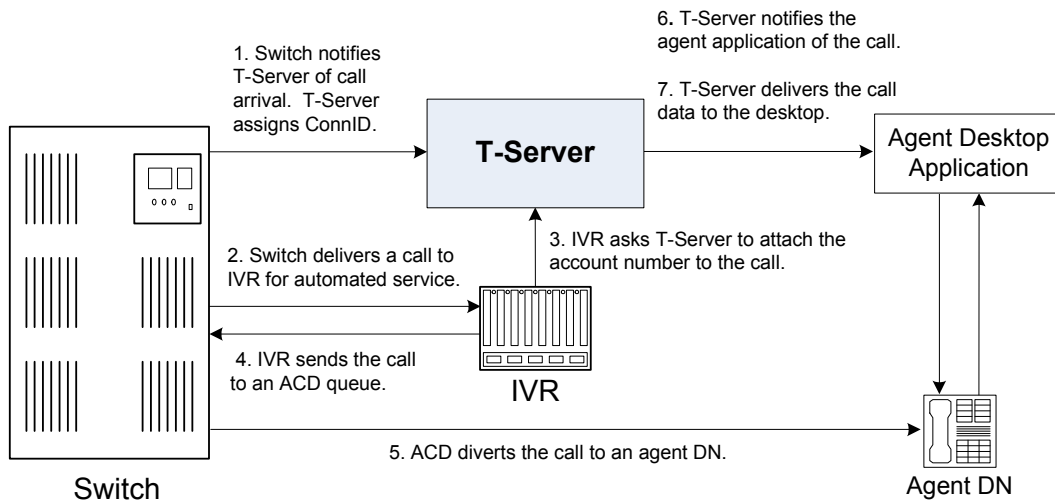


Figure 7: Functional T-Server Steps

Step 1

When the call arrives at the switch, T-Server creates a call in its internal structure. T-Server assigns the call a unique identifier, connection ID.

Step 2

The switch delivers the call to an Interactive Voice Response (IVR) unit, which begins automated interactions with the caller.

Step 3

IVR acquires user information from the caller through prompts and requests T-Server to attach that information to the call. T-Server updates the call with the user information.

Step 4

IVR sends the call to an ACD (Automated Call Distribution) queue.

Step 5

The ACD unit distributes the call to an available agent logged in to a particular DN (directory number).

Step 6

T-Server notifies the agent desktop application that the call is ringing on the agent DN. The notification event contains call data including ANI, DNIS, and account information that the IVR has collected.

Step 7

The agent desktop application presents the account information, including the name of the person whose account this is, on the agent's screen, so that the agent answering the call has all the relevant information.

These seven steps illustrate just a small part of T-Server's bridging, messaging, and interaction-processing capabilities.

Advanced Disconnect Detection Protocol

Since the 6.0 release of T-Server, the Advanced Disconnect Detection Protocol (ADDP) has replaced the Keep-Alive Protocol (KPL) as the method to detect failures for certain T-Server connections, including connections between two T-Servers and between a T-Server and its clients.

Notes: Starting with release 7.5, the KPL backward-compatibility feature is no longer supported.

ADDP applies only to connections between Genesys software components.

With ADDP, protocol activation and initialization is made on the client's side and you can change these parameters. No additional messages are sent when there is existing activity over the connection. T-Server client applications and the remote T-Server (if any) must be listening to the socket and respond promptly to the polling signal for the connection to be preserved.

If you are going to enable ADDP, you must do it using the [protocol](#), [addp-timeout](#), [addp-remote-timeout](#), and [addp-trace](#) configuration options. When configuring a timeout, consider the following issues:

- The configured timeout must be at least twice as long as the maximum network latency.
- There may be an interval when T-Server does not check for network activity.
- If the link connection fails but the client is not notified (for example, because the host is turned off, or because a network cable is unplugged), the maximum reaction time to a link-connection failure is equal to double the configured timeout plus the established network latency.

Also keep in mind that the T-Server receiving the polling signal may not respond immediately, and that a delay occurs after the polling signal, while the

response travels from one T-Server to another. If you do not account for these contingencies when configuring a timeout, the connection that ADDP is monitoring will be dropped periodically.

Redundant T-Servers

T-Servers can operate in a high-availability (HA) configuration, providing you with redundant systems. The basics of each T-Server's redundant capabilities differ from T-Server to T-Server. One basic principle of redundant T-Servers is the standby redundancy type, which dictates how quickly a backup T-Server steps in when the primary T-Server goes down.

The Framework Management Layer currently supports two types of redundant configurations: warm standby and hot standby. All T-Servers offer the warm standby redundancy type and, starting with release 7.1, the hot standby redundancy type is implemented in T-Servers for most types of switches. (See [Table 1](#).)

Instructions for configuring T-Server redundancy are available in Chapter 3, "High-Availability Configuration and Installation." Specifics on your T-Server's HA capabilities are outlined in Part Two of this document.

Note: IVR Server and some Network T-Servers can be configured for load sharing or warm or hot standby; however, they do not support any combination of these redundancy types. Details of your component's HA capabilities are discussed in Part Two of this document.

Support for Hot Standby Redundancy in Various T-Servers

Use [Table 1](#) to determine whether your T-Server supports the hot standby redundancy type. The table also indicates whether HA Proxy components are required for this support, and, if so, how many are required per pair of redundant T-Servers (or per link if so noted).

[Table 1](#) only summarizes hot standby redundancy support in various T-Servers. For detailed, up-to-date information on the subject, see the *Genesys Supported Media Interfaces Reference Manual* located on the Technical Support website at

<http://genesyslab.com/support/dl/retrieve/default.asp?item=A9CB309AF4DEB8127C5640A3C32445A7&view=item>.

Table 1: T-Server Support of the Hot Standby Redundancy Type

T-Server Type	Hot Standby Supported	HA Proxy Required	Number of HA Proxy Components
Aastra MXONE CSTA I	Yes	No	—
Alcatel A4200/OXO	Yes	No	—
Alcatel A4400/OXE	Yes	No	—
Aspect ACD	Yes	No	—
Avaya Communication Manager	Yes	No ^a	—
Avaya INDeX	Yes	No	—
Avaya TSAPI	Yes	No	—
Cisco UCCE	Yes	No	—
Cisco Unified Communications Manager	Yes	No	—
DataVoice Dharma	Yes	No	—
Digitro AXS/20	Yes	No	—
EADS Intecom M6880	Yes	No	—
EADS Telecom M6500	Yes	No	—
eOn eQueue	Yes	No	—
Fujitsu F9600	Yes	No	—
Huawei C&C08	Yes	No	—
Huawei NGN	Yes	No	—
Mitel MiTAI	Yes	No	—
NEC NEAX/APEX	Yes	No	—
Nortel Communication Server 2000/2100	Yes	Yes ^b , No ^c	1 per link
Nortel Communication Server 1000 with SCCS/MLS	Yes	No	—
Philips Sopho iS3000	Yes	No ^d	1
Radvision iContact	No	—	—
Samsung IP-PCX IAP	Yes	No	—

Table 1: T-Server Support of the Hot Standby Redundancy Type (Continued)

T-Server Type	Hot Standby Supported	HA Proxy Required	Number of HA Proxy Components
Siemens Hicom 300/HiPath 4000 CSTA I	Yes	No	—
Siemens HiPath 3000	Yes	No	—
Siemens HiPath 4000 CSTA III	Yes	No	—
Siemens HiPath DX	Yes	No	—
SIP Server	Yes	No	—
Spectrum	Yes	No	—
Tadiran Coral	Yes	No	—
Teltronics 20-20	Yes	Yes	1
Tenovis Integral 33/55	Yes	No	—
Network T-Servers^e			
AT&T	No	—	—
Concert	No	—	—
CRSP	No	—	—
DTAG	No	—	—
GenSpec	No	—	—
ISCP	No	—	—
IVR Server, using network configuration	Yes	—	—
KPN	No	—	—
MCI	No	—	—
NGSN	No	—	—
Network SIP Server	No	—	—
Sprint	No	—	—
SR3511	No	—	—
Stentor	No	—	—

- a. With release 7.1, T-Server for Avaya Communication Manager no longer uses HA Proxy for its support of hot standby. Earlier releases of this T-Server require two HA Proxies to support hot standby.
- b. For T-Server for Nortel Communication Server 2000/2100 in high-availability (hot standby) configuration, Genesys recommends that you use link version SCAI14 or above with call-progress and noncontroller-released messages enabled. See the switch-specific information in Part 2 of this *Deployment Guide* for additional information on HA configurations.
- c. Starting with release 7.5, T-Server for Nortel Communication Server 2000/2100 supports HA without HA Proxy when operating in Dual CTI Links mode. See the switch-specific information in Part 2 of this *Deployment Guide* for additional information on HA configurations.
- d. Starting with release 6.5.3, T-Server for Philips Sopho iS3000 supports HA both with and without HA Proxy.
- e. Although they do not support high availability per se, Network T-Servers do support a load-sharing schema.

Multi-Site Support

Multi-site configuration implies the existence of two or more switches that belong to the same enterprise or service provider, and that share the Genesys Configuration Database. (In some cases this may include isolated partitions on a given switch served by different T-Servers.) The main goal of T-Server support for multi-site operations is to maintain critical information about a call as it travels from one switch to another.

For instructions on installing and configuring a multi-site environment, including information on the Inter Server Call Control (ISCC) features, please see Chapter 3, “Multi-Site Support,” on [page 47](#).

Agent Reservation

T-Server provides support for clients to invoke the agent reservation function, `TReserveAgent()`. This function allows a server application that is a client of T-Server to reserve a DN along with an agent, a `Place`, or both, so that no other T-Server client can route calls to it during a specified reservation interval. Alternatively, when clients use the ISCC feature (see “ISCC Call Data Transfer Service” on [page 49](#)), they can use an agent reservation embedded in an ISCC request. (To do so, clients have to specify a certain `Extensions` attribute in an ISCC request when initiating an ISCC transaction. See [page 56](#) for the list of ISCC requests.)

The reservation does not currently prevent the reserved objects from receiving direct calls or calls distributed from ACD Queues; agent reservation is intended as a way of synchronizing the operation of several clients. See `RequestReserveAgent` in the *Voice Platform SDK 8.x .NET (or Java) API Reference* for more details on this function from the client’s point of view.

In addition to invoking the `TReserveAgent` function, you can customize the Agent Reservation feature by configuring options in the `T-Server Application` object. See “agent-reservation Section” on [page 190](#) in the “T-Server Common Configuration Options” chapter in Part Two for more details.

Starting with version 8.1, T-Server supports Agent Reservation failure optimization, to ensure that only agent reservation requests of the highest priority are collected. T-Server responds immediately with the `EventError` message to existing or new reservation requests of a lower priority while collecting the agent reservation requests of the highest priority only. This functionality is controlled with the `collect-lower-priority-requests` configuration option (see [page 190](#)).

Client Connections

The number of connections T-Server can accept from its clients depend on the operating system that T-Server runs. [Table 2](#) illustrates the number of client connections that T-Server support.

Table 2: Number of T-Server's Client Connections

Operating System	Number of Connections
AIX 32-bit mode (versions 5.3)	32767
AIX 64-bit mode (versions 5.3, 6.1, 7.1)	32767
HP-UX 32-bit mode (versions 11.11)	2048
HP-UX 64-bit mode (versions 11.11, 11i v2, 11i v3)	2048
HP-UX Itanium (version 11i v3)	2048
Linux 32-bit mode (versions RHEL 4.0, RHEL 5.0)	32768
Linux 64-bit mode (version RHEL 5.0)	32768
Solaris 32-bit mode (version 9)	4096
Solaris 64-bit mode (versions 9, 10)	65536
Windows Server 2003, 2008	4096



Chapter

3

Multi-Site Support

This chapter contains general information about multi-site environments, as well as information on deploying a multi-site environment for your T-Server.

This chapter is divided into the following sections:

- [Multi-Site Fundamentals, page 48](#)
- [ISCC Call Data Transfer Service, page 49](#)
- [ISCC/Call Overflow Feature, page 69](#)
- [Number Translation Feature, page 73](#)
- [Network Attended Transfer/Conference Feature, page 81](#)
- [Event Propagation Feature, page 83](#)
- [ISCC Transaction Monitoring Feature, page 90](#)
- [Configuring Multi-Site Support, page 90](#)

Note: Each switch/T-Server combination offers different multi-site options. For details describing your specific switch/T-Server environment, refer to Chapter 10, “T-Server Common Configuration Options,” on [page 181](#).

The following instructions apply to both local and remote switches and T-Servers. Because different vendor switches can be installed at the local and remote locations, this chapter covers several, but not all, possible configurations. To help determine which sections of this chapter apply to your situation, refer to Table 3 on [page 65](#) and Table 4 on [page 70](#).

For more information on your specific switch/T-Server environment, refer to the multi-site topics in Part Two of this guide.

Multi-Site Fundamentals

A multi-site configuration has two or more switches that belong to the same enterprise or service provider and that share the Genesys Configuration Database. (In some cases, this may include isolated partitions on a given switch served by different T-Servers.) The main goal of T-Server support for multi-site operations is to maintain critical information about a call as it travels from one switch to another.

T-Server supports multi-site operations using its *Inter Server Call Control* (ISCC; formerly called External Routing), which supports the following functions:

- **Call matching**—To link instances of a call distributed across multiple sites and to re-attach essential data associated with the call (ConnID, UserData, CallType, and CallHistory). The following T-Server features support this capability:
 - ISCC Call Data Transfer Service (active external routing)—when requested by a T-Server client by specifying the desired destination in the location parameter, and also with various ISCC strategies performed by direct dial or by using the Transfer Connect Service. See “ISCC Transaction Types” on [page 56](#) and “Transfer Connect Service Feature” on [page 68](#).
 - Inter Server Call Control/Call Overflow (ISCC/COF) feature (passive external routing)—applicable when calls are overflowed to another site either directly or manually (see [page 69](#)).
 - Number Translation feature (see [page 73](#)).
 - Network Attended Transfer/Conference (NAT/C) feature (see [page 81](#)).

Note: When ISCC detects call instance reappearance on a given site, the call is assigned a unique ConnID and the user data is synchronized with the previous call instances. This ensures that ConnIDs assigned to different instances of the same call on a given site are unique.

- **Call data synchronization between associated call instances (ISCC Event Propagation)**—To provide the most current data to call instances residing on remote T-Servers. The following T-Server features support this capability:
 - User Data propagation (see [page 84](#))
 - Party Events propagation (see [page 85](#))

Note: ISCC automatically detects topology loops and prevents continuous updates.

Note: In distributed networks, Genesys recommends using call flows that prevent call topology loops and multiple reappearances of the same call instance. This approach ensures that all T-Servers involved with the call report the same ConnID, and also optimizes telephony trunk allocation by preventing trunk tromboning.

The T-Server configuration contains information about other T-Servers with which it will communicate. T-Server uses this information to connect with the other T-Servers. During this “handshake” process, T-Servers exchange information about the following parameters:

- Protocol type
- Switch type
- Server name
- Location name (switch name)
- T-Server role (primary or backup)

To complete the handshake process, T-Servers exchange messages about the current condition of the links to their switches. After the handshake process is complete, T-Server is ready to support a multi-site operation.

ISCC Call Data Transfer Service

Because ISCC supports active external routing, T-Servers that serve different switches (usually on different sites) can exchange call data when a call is passed from one switch to another. With this functionality, T-Server provides its clients with the following additional information about each call received from another switch:

- The connection identifier of the call (attribute ConnID).
- Updates to user data attached to the call at the previous site (attribute UserData).
- The call type of the call (attribute CallType)—In multi-site environments the CallType of the call may be different for each of its different legs. For example, one T-Server may report a call as an Outbound or Consult call, but on the receiving end this call may be reported as Inbound.
- The call history (attribute CallHistory)—Information about transferring/routing of the call through a multi-site contact center network.

Note: Load-sharing IVR Servers and Network T-Servers cannot be designated as the destination location for ISCC, except when cast-type is set to dnis-pool. Consult the *Universal Routing Deployment Guide* for specific configuration details.

Figure 8 shows the steps that occur during a typical external routing (ISCC) transaction. Note that the location where a call is initially processed is called the *origination location*, and the location to which the call is passed is called the *destination location*.

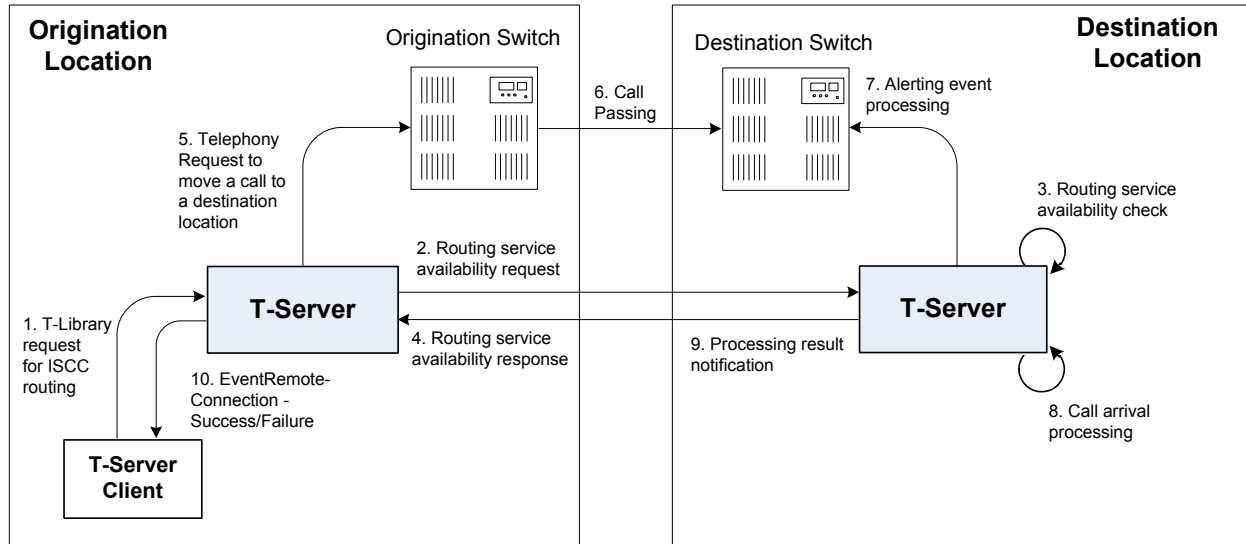


Figure 8: Steps in the ISCC Process

ISCC Call Flows

The following section identifies the steps (shown in Figure 8) that occur during an ISCC transfer of a call.

Step 1

A client connected to the T-Server at the origination location requests this T-Server to pass a call with call data to another location. For this purpose, the client must specify the `location` parameter (Attribute `Location`) when calling a corresponding T-Library function. ISCC processes the following T-Library requests:

- `TInitiateConference`
- `TInitiateTransfer`
- `TMakeCall`
- `TMuteTransfer`
- `TRouteCall`
- `TSingleStepTransfer`

Step 2

Upon receiving a client's request, the origination T-Server checks that the:

1. Connection to the destination T-Server is configured in the origination T-Server Properties dialog box.
2. The connection to the destination T-Server is active.
3. The destination T-Server is connected to its link.
4. The origination T-Server is connected to its link.

If these four conditions are met, the origination T-Server determines the transaction type that will be used for passing call data to another location in this transaction. The following possibilities exist:

- The client can request what *ISCC transaction type* (or simply *transaction type*) to use by specifying an appropriate key-value pair in the Extensions attribute of the request. The key-value pair must have a key equal to `iscc-xaction-type` and either an integer value as specified in the `TXRouteType` enumeration (see the *Voice Platform SDK 8.x .NET (or Java) API Reference*) or a string value equal to one of the following: `default`, `route`, `direct` (or `direct-callid`), `direct-network-callid`, `direct-notoken`, `direct-ani`, `direct-uu`, `direct-digits`, `reroute`, `dnis-pool`, `pullback`, or `route-uu`.
- If the client does not specify the transaction type in the request or specifies the `default` transaction type, T-Server checks the Switch configuration for the transaction type configured in the Access Code (or Default Access Code) properties:
 - If the Route Type property of the Access Code is set to any value other than `default`, T-Server uses the specified value as the transaction type.
 - If the Route Type property of the Access Code is set to the `default` value, T-Server uses the first value from the list specified in the `cast-type` configuration option configured for the destination T-Server. If no value has been specified for the `cast-type` option, the default value of `route` is used as the transaction type.

Note: For more information on Access Codes and Default Access Code, see “Switches and Access Codes” on [page 92](#).

After the origination T-Server determines the requested transaction type, it determines if the destination T-Server supports this transaction type.

You must list the transaction types T-Server supports in the `cast-type` configuration option.

The origination T-Server issues a request for routing service availability and sends it to the destination T-Server. The T-Server request contains data that should be passed along with the call to the destination location. This data includes the transaction type, ConnID, UserData, CallType, and CallHistory.

The timer specified by the `request-tout` configuration option is set when the origination T-Server sends the request. If either the specified timeout expires or the call is abandoned before the origination T-Server receives a response from the destination T-Server, the operation is considered failed. In this scenario, the origination T-Server:

1. Generates a request to the destination T-Server to cancel the request for routing service.
2. Sends `EventError` to the client that requested the service.
3. Deletes information about the request.

Step 3

The destination T-Server receives the request for routing service availability and checks the requested type of routing. Depending on the ISCC transaction type, it stores the request information and, when appropriate, allocates access resources for the coming call. For example, an External Routing Point is allocated when the transaction type is `route`, and an Access Resource of type `dnis` is allocated when the transaction type is `dnis-pool`.

Note: The `resource-allocation-mode` and `resource-load-maximum` configuration options determine how resources are allocated. For option descriptions, refer to Chapter 10, “T-Server Common Configuration Options,” on [page 181](#) for option descriptions.

If resources are unavailable, the request is queued at the destination location until a resource is free or the origination T-Server cancels the request. If the request is canceled, the destination T-Server deletes all information about the request.

If resources are unavailable because of incorrect configuration, the destination T-Server returns an error event to the origination T-Server.

Step 4

If resources are available, the destination T-Server generates a positive response and the timer is started for the interval specified by the `timeout` configuration option of the destination T-Server.

Step 5

If the origination T-Server receives a negative response, it sends an `EventError` message to the client and clears all data about the request.

If the origination T-Server receives the confirmation about routing service availability, it processes the client’s request and sends a corresponding message to the switch. The timer on the origination T-Server is also started for the interval specified by the `timeout` configuration option of the destination T-Server.

Step 6

The origination switch processes the T-Server request and passes the call to the destination switch.

Step 7

If the call arrives at the destination switch, the switch generates an alerting event.

The destination T-Server waits for the call no longer than the interval specified by the timeout configured on the destination T-Server. If the call is not received at the destination location within this interval, the destination T-Server issues a failure notification to the origination T-Server, deletes all data about the request, and, when appropriate, frees the resources previously allocated for the request.

If either the specified timeout expires or the call is abandoned before the origination T-Server receives a response from the destination T-Server, the operation is considered failed. In this case, the origination T-Server:

1. Generates a request to the destination T-Server to cancel the request for routing service.
2. Responds to the client that requested the service in one of the following ways:
 - If the origination T-Server has already sent a response to the request the client sent in Step 1, the origination T-Server supplements its response with `EventRemoteConnectionFailed`.
 - If the origination T-Server has not yet sent a response to the client, the origination T-Server sends `EventError`.
3. Deletes information about the request.

Step 8

If the destination T-Server matches the arrived call, it updates the `ConnID`, `UserData`, `CallType`, and `CallHistory` attributes with the data received in the request for routing service availability. The connection ID is updated as follows:

The arrived call is assigned the `ConnID` that is specified in the request for routing service availability, but only if this `ConnID` does not coincide with the `ConnID` of a call that has existed at the destination site. If two such `ConnIDs` are identical, the arrived call is assigned a new unique `ConnID`.

For `direct-*` transaction types (where the asterisk stands for a `callid`, `uui`, `ani`, or `digits` extension), the call reaches the destination DN directly.

For the transaction types `route` and `route-uui`, the call first arrives at an External Routing Point from which it is routed to the destination DN. The call info is updated when the call reaches the External Routing Point. An External

Routing Point is considered free when the first alerting event (`EventQueued` or `EventRouteRequest`) is distributed.

Please keep the following issues in mind when using the ISCC feature:

- If routing from a dedicated External Routing Point to the destination DN fails, T-Server considers the transaction failed. However, the `ConnID`, `UserData`, `CallType`, and `CallHistory` attributes are updated. Then, T-Server attempts to route the call to one of the Default DNs configured for this External Routing Point.
- If the destination T-Server did not receive a request for routing service availability, but a call arrives at an External Routing Point, T-Server considers the call to be unexpected and routes the call to the DN specified by the `dn-for-unexpected-calls` configuration option. When no alternative targets are defined, the call remains at the External Routing Point until diverted by the switch or abandoned by the caller.

For `reroute` and `pullback` transaction types, the call returns to the network location. For the `dnis-pool` transaction type, the call reaches the destination DN directly.

Step 9

If, in Step 8, the call does not arrive within the configured timeout, or the transaction fails, the destination T-Server sends a notification of failure to the origination T-Server.

Otherwise, the destination T-Server notifies the origination T-Server that the routing service was successful and deletes all information about the request.

Step 10

The origination T-Server notifies the client that the routing service was successful (or failed) and deletes all information about the request.

Client-Controlled ISCC Call Flow

The following section identifies the steps that occur during a client-controlled ISCC transfer of a call.

Step 1

A client, such as Universal Routing Server (URS), that is connected to the T-Server at the origination location detects a call to be delivered to another destination location.

Step 2

The client chooses a destination location and the target DN for the call. Then, it sends the `TGetAccessNumber` request to the destination T-Server for routing service availability, indicating the target DN and other call context (`ConnID`, `UserData`, and `CallHistory` attributes).

Step 3

The destination T-Server receives the request for routing service availability. Depending on the ISCC transaction type, it stores the request information, including the call context. When appropriate, it allocates access resources for the coming call, such as External Routing Point.

If resources are unavailable, the request is queued at the destination T-Server until an appropriate ISCC resource is free or the client cancels the request. If the request is canceled, the destination T-Server deletes all information about the request.

If resources are unavailable because of incorrect configuration, the destination T-Server returns an `EventError` message to the client.

Step 4

The destination T-Server replies to the client with the `EventAnswerAccessNumber` message, which contains the allocated ISCC resource.

Step 5

The client requests that the origination T-Server delivers the call to the destination location using the allocated access resource.

Step 6

The origination T-Server receives and processes the client's request, and then sends a corresponding message to the switch.

Step 7

The call arrives at the destination switch and is reported to the destination T-Server via CTI. The call is matched by means of ISCC, based on the specified `cast-type` setting and allocated resource, and then the call is assigned a requested call context (such as `ConnID` or call data). Upon successful transaction completion, the destination T-Server notifies the client by sending `EventRemoteConnectionSuccess`.

The destination T-Server waits for the call no longer than the interval specified by the timeout that is configured on the destination T-Server. If the call is not received at the destination location within this interval, the destination T-Server issues a failure notification to the client by sending

`EventRemoteConnectionFailed`, deletes all data about the request, and, when appropriate, frees the resources previously allocated for the request.

The destination T-Server notifies the client whether the routing service succeeded or failed by sending either the `EventRemoteConnectionSuccess` or `EventRemoteConnectionFailure`, respectively.

ISCC Transaction Types

As switches of different types provide calls with different sets of information parameters, a single mechanism for passing call data between the switches is not feasible in some cases. Therefore, the ISCC feature supports a number of mechanisms for passing call data along with calls between locations. This section describes ISCC transaction type principles, identifies which transaction types are supported for each T-Server, and defines each transaction type (beginning with “direct-ani” on [page 57](#)).

It is important to distinguish the two roles that T-Servers play in an external routing (ISCC) transaction—namely *origination T-Server* and *destination T-Server*:

- The origination T-Server initiates an ISCC transaction. It prepares to send the call to another T-Server and coordinates the process.
- The destination T-Server receives call data from an origination T-Server and matches this data to a call that will arrive at some time in the future.

The distinction between these roles is important because the range of telephony-hardware functionality often requires T-Servers to support two entirely different sets of ISCC transactions based on which of the two roles they play. For instance, it is very common for a particular T-Server to support many types of ISCC transactions when it takes on the origination role, but fewer when it takes on the role of a destination T-Server.

The ISCC transaction type `reroute` is a good example. Most T-Servers support `Reroute` as origination T-Servers, but very few support `Reroute` as destination T-Servers.

Determining and Configuring Transaction Type Support

You can find descriptions of these transaction types starting on [page 57](#). Use Table 3 on [page 65](#) to identify the transaction types your destination T-Server supports. A blank table cell indicates that T-Server does not support a certain transaction type.

You can configure the transaction types specific to your T-Server as values of the `cast-type` configuration option specified in the ISCC configuration section `extrouter`. Refer to Chapter 10, “T-Server Common Configuration Options,” on [page 181](#) for the option description.

ISCC Transaction Type General Principles

Generally, since most of the ISCC implementation is done at the T-Server Common Part (TSCP) code level, all T-Servers support certain ISCC transaction types. Any T-Server can act as the origination T-Server for the following transaction types:

- `direct-ani`, [page 57](#)
- `direct-notoken`, [page 59](#)
- `dnis-pool`, [page 60](#)
- `pullback`, [page 61](#)
- `reroute`, [page 62](#)
- `route` (aliased as `route-notoken`), the default transaction type, [page 63](#)

The following transaction types are unevenly supported for both the origination and destination T-Server roles:

- `direct-callid` (aliased as `direct`), [page 58](#)
- `direct-digits` (reserved for Genesys Engineering)
- `direct-network-callid`, [page 58](#)
- `direct-uui`, [page 59](#)
- `route-uui`, [page 64](#)

The `reroute` and `pullback` transaction types are supported only for selected T-Servers in the *destination* role. However, if you implement this support, other transaction types require additional configuration and testing—even those that would normally be supported by default.

direct-ani

With the transaction type `direct-ani`, the ANI call attribute is taken as the parameter for call matching. Properly configured switches and trunks can keep the ANI attribute when a call is transferred over the network. T-Server can use this network feature for call matching.

Warning! Depending on the switch platform, it may be possible to inherit the ANI attribute after routing a call to a remote destination, and after performing a single-step transfer and other telephone actions. However, ISCC only works properly in scenarios where the ANI attribute on the destination T-Server is represented by exactly the same digit string as on the origination T-Server.

Typically, the ANI attribute represents the original call identifier (customer phone number), which guarantees that the attribute remains unique. However, you can use the `non-unique-ani` resource type to block ISCC from matching calls based on an ANI that is known to be non-unique. (See “Configuring access resources for non-unique ANI” on [page 101](#) for details.)

direct-callid

With the transaction type `direct-callid`, the call reaches the destination DN directly from another location, and the `CallID` of the call is taken as the attribute for call matching. When a call arrives at the final destination, the destination T-Server identifies its `CallID`, and updates the call info if the `CallID` matches.

Use this transaction type when the destination switch has the capability to assign to an incoming call the same network-wide unique `CallID` that the origination switch has already assigned to that call.

Notes: The `direct-callid` transaction type is used only in conjunction with the `TRouteCall` and `TSingleStepTransfer` function calls. It is applied only to the call that is in progress, and does not apply to functions that involve in the creation of a new call, such as `TMakeCall`.

For T-Server for Nortel Communication Server 2000/2100, the `direct-callid` transaction type is also applied to the `TMuteTransfer` function.

direct-network-callid

With the transaction type `direct-network-callid`, the call reaches the destination DN directly from another location, and the `NetworkCallID` of the call is taken as the attribute for call matching. When a call arrives at the final destination, the destination T-Server identifies its `NetworkCallID`, and updates the call info if the `NetworkCallID` matches.

Use this transaction type when the destination switch has the capability to assign to an incoming call the same network-wide unique `NetworkCallID` that the origination switch has already assigned to that call.

Note: To support this transaction type, you must configure `Target Type` and `ISCC Protocol Parameters` fields of the corresponding `Switch Access Code` in the Configuration Layer. For information about settings that are specific for your T-Server type, refer to Part Two of this document.

direct-uui

With the transaction type `direct-uui`, so-called user-to-user information (UUI) is taken as the attribute for call matching. Some switches make it possible to send a small data packet along with a call. T-Server can use this data to recognize a call passed from one switch to another. The destination T-Server generates a local unique value for UUI, and then notifies the origination T-Server. The origination T-Server uses a provided value to mark the call coming from the origination location. The destination T-Server receives a call and checks whether it is marked with an exact UUI value. If so, the call is considered to be matched.

On the Avaya Communication Manager and the Aspect ACD, UUI is referred to as “user-to-user information.” On the Siemens Hicom 300 switch with CallBridge, UUI is referred to as “Private User Data.” On the Alcatel A4400/OXE switch, UUI is referred to as “correlator data.”

Note: To support this transaction type, you must configure your switches to pass the UUI provided by your T-Server. You must also ensure that the trunks involved do not drop this data.

direct-notoken

With the transaction type `direct-notoken`, T-Server expects a call to arrive from another location to the destination DN specified in the request for routing service availability. When a call reaches the specified DN, T-Server processes the call as the expected externally-routed call.

Notes: This matching criterion is weak because any call that reaches the specified DN is considered to be the expected call. Genesys recommends that you use this transaction type only in a contact center subdivision that can only be reached from within the contact center (such as the second line of support, which customers cannot contact directly).

When using direct transaction types, Network T-Servers and load-sharing IVR Servers are not meant to act as destination T-Servers for call routing. Using Network T-Server with these transaction types requires special architecture.

dnis-pool

With the `dnis-pool` transaction type, T-Server reserves one of its DNIS access resources and waits for the call that has the same DNIS attribute as the name of the reserved DNIS access resource.

If the arrived call is matched successfully, the destination T-Server may update the value of the DNIS attribute of the call (along with `ConnID`, `UserData`, `CallType`, and `CallHistory`) with the value of the DNIS attribute of the original call. This occurs when the value of the DNIS attribute of the original call is specified as a value of the key-value pair `_ISCC_TRACKING_NUMBER_` in the `Extensions` attribute of the original client request.

The DNIS matching can be based on any number of digits out of all the digits that comprise the DNIS attribute. The number of digits that T-Server should use for DNIS matching is specified for the destination switch as the `ISCC Protocol Parameters` property of the Switch Access Code. The value syntax should be as follows:

`dnis-tail=<number-of-digits>`

For example, if this property is set to the `dnis-tail=7` value, ISCC matches only the last seven digits of a DNIS.

You must configure DNIS access resources in the switch; otherwise, ISCC fails to use this transaction type and sends `EventError` in response to the client application request.

Note: The `dnis-pool` transaction type is typically used for networks that employ a “behind the SCP” architecture, such as network IVR. Network T-Server for GenSpec and IServer are two examples of this, but other Network T-Servers might also be used in this architecture.

In Load-Balancing Mode

When T-Server uses load balancing for call routing with the `dnis-pool` transaction type, the following processes occur:

1. A client of the origination T-Server sends a request to pass a call to the location with a DNIS access resource specified in the key-value pair `iscc-selected-dnis`.
2. The origination T-Server distributes the request for a routing service to all destination T-Servers.
3. The destination T-Servers receive the request and check that the specified DNIS is not being used by another routing service request.
4. The origination T-Server expects to receive a positive response from each destination T-Server. If the origination T-Server receives a negative response from at least one T-Server, it sends an `EventError` to the client and clears all data about the request. If the origination T-Server receives the confirmation about routing service availability from all destination T-Servers, it processes the client's request and sends a corresponding message to the switch.
5. The origination switch processes the T-Server request and passes the call to the destination switch.
6. The call arrives at the destination switch, which generates an alerting event to one of the corresponding load-balanced destination T-Servers.
7. That destination T-Server processes the call and notifies the origination T-Server that the routing service was successful and deletes all information about the request.
8. The origination T-Server sends a routing service request cancellation to all other destination T-Servers.
9. The origination T-Server notifies the client that the routing service has been successful and deletes all information about the request.

pullback

`PULLBACK` is used in the following scenario, for those T-Servers that support it:

1. A call arrives at Site A served by a Network T-Server.
2. At Site A, a Network T-Server client requests to pass the call by means of ISCC routing to Site B served by a premise T-Server. Any transaction type except `reroute` or `pullback` can be specified in this request.
3. The call arrives at Site B and is either answered by an agent or delivered to a routing point.
4. A client of the premise T-Server at Site B sends a `TRouteCall` or `TSingleStepTransfer` request to transfer the call to the network.

5. The Site B premise T-Server notifies the Network T-Server about this request.
6. The network T-Server receives the notification and issues an `EventRouteRequest` to obtain a new destination.
7. After receiving the new destination information, the Network T-Server disconnects the call from its current premise location at Site B and attempts to route the call to the new destination.
8. The Site B premise T-Server stops tracking the call, which has disconnected from the premise's agent DN or routing point and is delivered to the network.
9. The network T-Server completes routing the call to its new destination.

Note: The transaction type `pullback` can only be used to return a call from a premise T-Server to the Network T-Server that serves the site from which the call was previously transferred.

reroute

`Reroute` is used in the following scenario, for those T-Servers that support it:

1. A call arrives at Site A served by a Network T-Server.
2. At Site A, a Network T-Server client requests to pass the call by means of ISCC to Site B served by a premise T-Server. Any transaction type except `reroute` or `pullback` can be specified in this request.
3. An agent at Site B answers the call.
4. A client of the premise T-Server at Site B sends a `TSingleStepTransfer` or `TRouteCall` request to transfer the call elsewhere (to a PSTN, to an agent, or to a routing point).
5. The Site B premise T-Server notifies the Network T-Server about this request and releases the call leg that resides at the agent's phone (using `TReleaseCall`) or at the Routing Point (using `TRouteCall` with the parameter `RouteTypeCallDisconnect`).
6. The Network T-Server receives the notification and reroutes the call to the requested destination by sending `EventRouteRequest` and attaching the call's user data.

Notes: The transaction type `reroute` can only be used to return a call from a premise T-Server to the Network T-Server that serves the site from which the call was previously transferred.

To perform multi-site operations that are initiated with `TRouteCall` and for which the `reroute` transaction type is requested, the origination T-Server must support the `RouteTypeCallDisconnect` subtype of `TRouteCall`.

route

With the transaction type `route` (aliased as `route-notoken`), a call from the origination location reaches a dedicated External Routing Point, and from there, it is routed to a destination DN.

To control configured External Routing Points, T-Server must register these DNs with the switch. Failure to register implies that the External Routing Point is not available for ISCC purposes. Client applications can register External Routing Points via T-Server for monitoring purposes only.

Point-to-Point (One-to-One)

In the Point-to-Point access mode, only one trunk line is used to access an External Routing Point (for example, VDN, CDN) at the destination site. See [Figure 9](#).

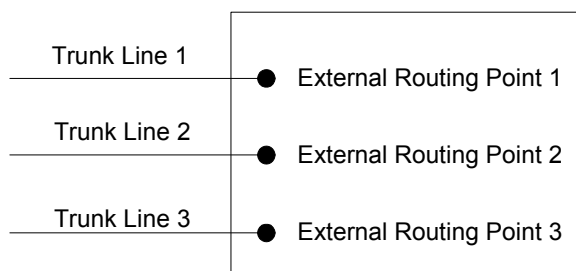


Figure 9: Point-to-Point Trunk Configuration

Note: Dedicated DNs of the External Routing Point type must be configured in a switch. See “Configuring Multi-Site Support” on [page 90](#).

Multiple-to-Point (Multiple-to-One)

In the Multiple-to-Point access mode, trunk lines are assigned to the destination switch’s trunk group, from which calls are routed to the final destination. See [Figure 10](#).

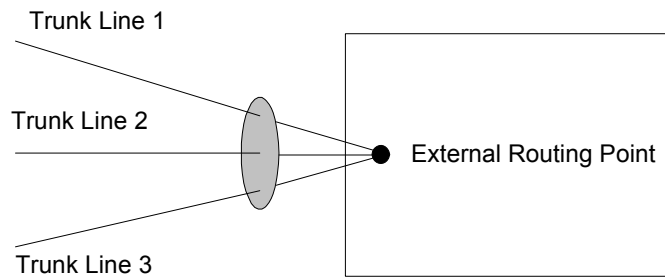


Figure 10: Multiple-to-Point Trunk Configuration

With this configuration, all calls reach the same External Routing Point. The DNIS attribute of a specific call differs from that of other calls and uniquely identifies the trunk from which the call arrived.

Note: To switch to this operating mode, you must configure the `route-dn` configuration option for T-Server.

route-uui

The `route-uui` transaction type employs the dedicated External Routing Point feature of the `route` transaction type (page 63) and the UUI matching feature of the `direct-uui` transaction type (page 59). This transaction type accommodates those switches that require a designated External Routing Point even though they use UUI for tracking.

Note: To support this transaction type, you must configure your switches to pass the UUI provided by your T-Server. You must also ensure that the trunks involved do not drop this data.

T-Server Transaction Type Support

Table 3 shows which transaction types are supported by a specific T-Server. Use this table to determine the transaction types that are available for use with your T-Server. This applies both to the `cast-type` you specify in the configuration options for your T-Server, and to any client-designated route-type requests specified for transfers of calls. A blank table cell indicates that T-Server does not support a certain transaction type.

Table 3: T-Server Support of Transaction Types

T-Server Type	Transaction Type										
	route		re-route	direct-callid	direct-uuui / route-uuui	direct-no-token	direct-ani	direct-digits	direct-network-callid	dnis-pool	pull-back
	one-to-one	multiple-to-one									
Aastra MXONE CSTA I	Yes			Yes ^a		Yes	Yes ^a				
Alcatel A4200/OXO	Yes			Yes		Yes	Yes				
Alcatel A4400/OXE	Yes			Yes ^{a,b,c}	Yes ^d	Yes	Yes ^a		Yes ^e		
Aspect ACD	Yes	Yes		Yes ^c		Yes ^f	Yes ^f				
Avaya Communication Manager	Yes				Yes	Yes	Yes				
Avaya INDeX	Yes					Yes	Yes ^b				
Avaya TSAPI	Yes				Yes	Yes	Yes				
Cisco UCCE	Yes					Yes	Yes				
Cisco Unified Communications Manager	Yes			Yes		Yes	Yes				
DataVoice Dharma	Yes			Yes		Yes	Yes				
Digitro AXS/20	Yes			Yes		Yes					
EADS Intecom M6880	Yes			Yes		Yes	Yes				
EADS Telecom M6500	Yes			Yes		Yes	Yes				
eOn eQueue	Yes			Yes		Yes					
Fujitsu F9600	Yes					Yes					

Table 3: T-Server Support of Transaction Types (Continued)

T-Server Type	Transaction Type										
	route		re-route	direct-callid	direct- uui / route- uui	direct- no- token	direct- ani	direct- digits	direct- network- callid	dnis- pool	pull- back
	one-to- one	multiple- to-one									
Huawei C&C08	Yes			Yes							
Huawei NGN	Yes					Yes	Yes				
Mitel MiTAI	Yes					Yes	Yes		Yes ^g		
NEC NEAX/APEX	Yes			Yes		Yes	Yes				
Nortel Communication Server 2000/2100	Yes			Yes ^f		Yes ^f	Yes ^f				
Nortel Communication Server 1000 with SCCS/MLS	Yes			Yes		Yes	Yes		Yes		
Philips Sopho iS3000	Yes			Yes		Yes	Yes				
Radvision iContact	Yes		Yes								Yes
Samsung IP-PCX IAP	Yes			Yes		Yes					
Siemens Hicom 300/HiPath 4000 CSTA I	Yes			Yes	Yes ^d	Yes	Yes				
Siemens HiPath 3000	Yes			Yes		Yes					
Siemens HiPath 4000 CSTA III	Yes				Yes ^d	Yes	Yes				

Table 3: T-Server Support of Transaction Types (Continued)

T-Server Type	Transaction Type										
	route		re-route	direct-callid	direct-uuui / route-uuui	direct-no-token	direct-ani	direct-digits	direct-network-callid	dnis-pool	pull-back
	one-to-one	multiple-to-one									
Siemens HiPath DX	Yes				Yes ^h	Yes	Yes ⁱ				
SIP Server	Yes		Yes		Yes ^j	Yes					Yes
Spectrum	Yes	Yes		Yes		Yes ^f	Yes ^f				
Tadiran Coral	Yes			Yes		Yes	Yes				
Teltronics 20-20	Yes			Yes		Yes	Yes				
Tenovis Integral 33/55	Yes			Yes		Yes	Yes				
Network T-Servers											
AT&T											
Concert											
CRSP											Yes
DTAG			Yes								
GenSpec	Yes	Yes	Yes							Yes	
IVR Server, using network configuration	Yes	Yes	Yes							Yes	Yes
KPN			Yes								
ISCP											
MCI											
NGSN	Yes									Yes	Yes
Network SIP Server	Yes					Yes	Yes			Yes	
Sprint	Yes										

Table 3: T-Server Support of Transaction Types (Continued)

T-Server Type	Transaction Type										
	route		re-route	direct-callid	direct-uuui / route-uuui	direct-no-token	direct-ani	direct-digits	direct-network-callid	dnis-pool	pull-back
	one-to-one	multiple-to-one									
SR-3511											
Stentor											

- a. Not supported in the case of function `TRouteCall` on a Virtual Routing Point: a Routing Point can be simulated using a hunt group with calls being deflected or transferred from the hunt-group member when routing. When a two-step (typically mute) transfer is used on such a hunt-group member, `CallID` and `ANI` usually change; thus, the `direct-callid` and `direct-ani` types do not work.
- b. Not supported in the case of function `TSingleStepTransfer` when the T-Server service is simulated using a two-step transfer to the switch. In this case, `CallID` and `ANI` change; thus, the `direct-callid` and `direct-ani` types do not work.
- c. Not supported if two T-Servers are connected to different nodes.
- d. There are some switch-specific limitations when assigning CSTA correlator data `UUUI` to a call.
- e. Supported only on ABCF trunks (Alcatel internal network).
- f. To use this transaction type, you must select the `Use Override` check box on the Advanced tab of the DN Properties dialog box.
- g. Supported only for `TRouteCall` requests made from a Native Routing Point.
- h. Not supported if a `TMakeCall` request is made.
- i. Not supported if a `TInitiateTransfer` or `TInitiateConference` request is made from an outgoing call on a device.
- j. SIP Server supports the `direct-uuui` type.

Transfer Connect Service Feature

The Transfer Connect Service (TCS) feature supports transfer connect services available on some telephony networks. When this feature is enabled, ISCC passes user data to remote locations to which calls are transferred or conferenced using transfer connect services.

Procedure: Activating Transfer Connect Service

Start of procedure

1. Open the T-Server Application's Properties dialog box.
2. Click the Options tab.
3. Set the `tcs-use` configuration option to always.
4. Set the `tcs-queue` configuration option to the number of a DN on the origination switch.

ISCC uses this DN as an intermediate step when sending calls to the remote location. The DN that is configured as `tcs-queue` receives attached data indicating the Feature Access Code (FAC) needed to reach the remote site. After a call is directed to the DN with data, a monitoring application takes the data and generates the required DTMF (dual-tone multifrequency) tones to redirect the call through the network to the remote location.

5. When you are finished, click Apply.
6. Click OK to save your changes and exit the Properties dialog box.

End of procedure

Note: With T-Server for Avaya Communication Manager, you can use `RequestRouteCall` with `RouteTypeOverwriteDNIS` to initiate the playing of DTMF tones. This is done through the use of another intermediate DN (typically, an announcement port configured to give the silent treatment), to which the call is routed. When the call is established on this DN, T-Server requests that the digits sent in the DNIS field of the `TRouteCall` be played by using the `ASAI-send-DTMF-single` procedure.

ISCC/Call Overflow Feature

The Inter Server Call Control/Call Overflow (ISCC/COF) feature of T-Server, that supports *passive external routing*, is specifically designed to handle calls delivered between sites without an explicitly defined destination location. Such scenarios include contact center overflows and manual call transfers.

An *overflow situation* occurs when a call comes into a contact center where all agents are currently busy. In this situation, the switch can transfer (overflow) the incoming call to another site where there is an available agent.

T-Server uses two methods to handle call overflow and manual transfer scenarios. The first method is based on `NetworkCallID` matching and the second method is based on `ANI/OtherDN` matching.

When connected to each other via switch-specific networks, switches of some types can pass additional information along with transferred calls. This information may contain the `NetworkCallID` of a call, which is a networkwide unique identifier of the call.

When connected via a regular PSTN, switches of all types can send the `ANI` and/or `OtherDN` attributes to the destination switch during any call transfer operation.

While all T-Servers support the ISCC/COF feature using the `ANI` and/or `OtherDN` attributes, only a few support this feature using the `NetworkCallID` attribute. Table 4 shows the T-Server types that provide the `NetworkCallID` of a call.

Table 4: T-Server Support of NetworkCallID for ISCC/COF Feature

T-Server Type	Supported NetworkCallID Attribute
Alcatel A4400/OXE ^a	Yes
Aspect ACD	Yes
Avaya Communication Manager ^{a,b}	Yes
Avaya TSAPI ^{a,b}	Yes
Cisco UCCE	Yes
Mitel MiTAI ^a	Yes
Nortel Communication Server 2000/2100 ^a	Yes
Nortel Communication Server 1000 with SCCS/MLS ^a	Yes
SIP Server ^a	Yes
Spectrum	Yes

a. Supported only if the `match-flexible` configuration parameter is used.

b. ISCC/COF is cross-compatible between T-Server for Avaya Communication Manager and T-Server for Avaya TSAPI.

The ISCC/COF feature can use any of the three attributes (`NetworkCallID`, `ANI`, or `OtherDN`) as criteria for matching the arriving call with an existing call at another location. Consequently, the attribute that is used determines what

ConnID, UserData, CallType, and CallHistory are received for the matched call from the call's previous location.

Warning! Depending on the switch platform, it may be possible to inherit the ANI attribute after routing a call to a remote destination, and after performing a single-step transfer and other telephone actions. However, ISCC/COF works properly only in scenarios where the ANI attribute on the destination T-Server is represented by exactly the same unique digit string as on the origination T-Server.

Typically, the ANI attribute represents the original call identifier (customer phone number), which guarantees that the attribute remains unique.

Note: When the ISCC/COF feature is in use, the Number Translation feature becomes active. For more information on feature configuration, see “Number Translation Feature” on [page 73](#).

ISCC/COF Call Flow

[Figure 11](#) shows the sequence of steps that occur in an ISCC/COF scenario when a call is made or transferred by an agent at Site A to a DN at Site B, or when a call is overflowed from Site A to Site B.

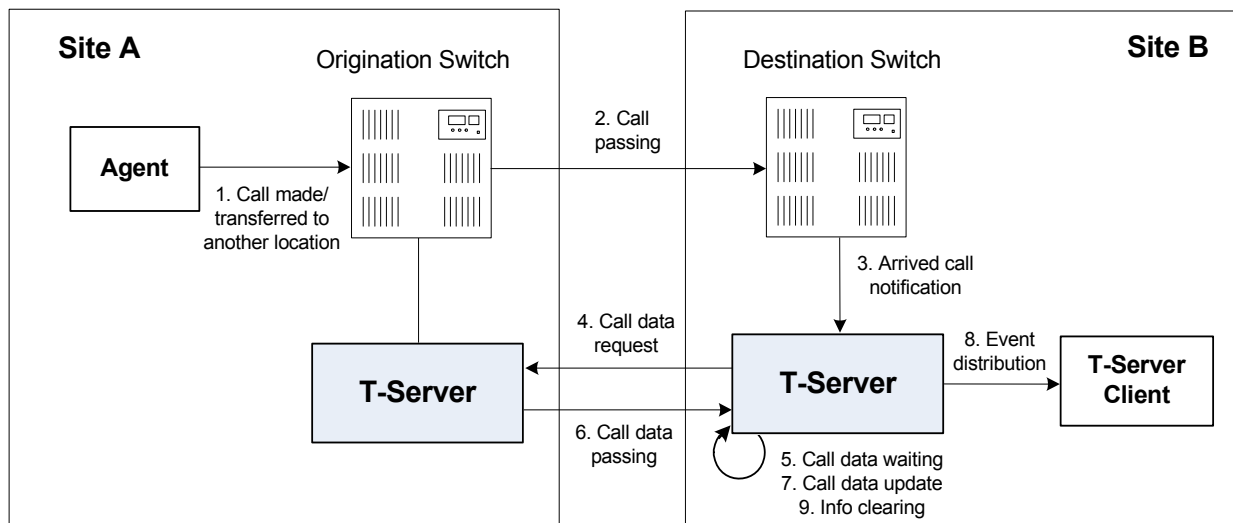


Figure 11: Steps in the ISCC/COF Process

Step 1

An agent makes or transfers a call manually to another location or a call is overflowed from Site A (origination location) to Site B (destination location).

Step 2

Switch A (the origination switch) passes the call to Switch B (the destination switch).

Step 3

The call reaches the destination switch, which notifies the destination T-Server about the arrived call.

Step 4

The destination T-Server verifies with remote locations whether the call overflowed at any of them.

To determine which calls to check as possibly having overflowed, T-Server relies on the Switch object and the presence of DNs on the Switch configured as the Access Resource type with the Resource Type set either to `cof-in` (COF-IN DNs) or to `cof-not-in` (COF-NOT-IN DNs):

T-Server skips an arriving call when one of following conditions is met:

- The call arrives at a DN configured as an Enabled COF-NOT-IN DN.
- COF-IN DNs are configured, but the call arrives at a DN other than one of the configured COF-IN DNs or to a COF-IN DN which is Disabled.

In all other cases, the call is checked for overflow.

To determine which location the call arrived from, T-Server checks the call type and checks whether the call has the `NetworkCallID`, `ANI`, or `OtherDN` attribute:

- If the call is not an inbound call, the request for call data is sent to all remote locations *except* those whose Switch Access Code has the ISCC Call Overflow Parameters property set to `inbound-only=true`.
- If the call of any type has the `NetworkCallID` attribute, the destination T-Server sends a request for call data to the remote locations of the same switch type as the destination location if their Switch Access Codes have the ISCC Call Overflow Parameters property set to `match-callid`.
- If the call of any type has the `ANI` or `OtherDN` attribute, the request for call data is sent to remote locations whose Switch Access Code has the ISCC Call Overflow Parameters property set to `match-ani`.

Step 5

The destination T-Server waits (suspending events related to that call) for the call data from the remote T-Server for the time interval specified in the `cof-ci-req-tout` configuration option. Within this interval, T-Server holds any events related to the call. In addition, the `cof-ci-defer-delete` option on the origination T-Server establishes the time interval only after which that T-Server deletes the call information. And the `cof-ci-wait-all`, if set to true,

forces the origination T-Server to wait for responses related to possible call overflow situations before updating call data.

Step 6

The T-Server at the location from which the call was transferred or overflowed sends call data to the requesting T-Server.

Step 7

If a positive response to the call-data request is received, T-Server updates ConnID, UserData, CallType, and CallHistory, distributes all suspended events related to that call, and deletes all information regarding the transaction (Step 9).

Step 8

If the timeout set by `cof-ci-req-tout` expires, T-Server distributes all suspended events, and starts the timeout specified by the `cof-rci-tout` option. If a positive response is received within the timeout set by `cof-rci-tout`, T-Server updates the ConnID, UserData, CallType, and CallHistory, and notifies client applications by distributing `EventPartyChanged`.

Step 9

T-Server deletes all information regarding the transaction when one of these results occurs:

- The first positive response to the call-data request is received.
- Negative responses from all queried locations are received.
- The timeout specified by the `cof-rci-tout` option expires.

Number Translation Feature

The Number Translation feature of T-Server extends the ISCC/COF and `direct-ani` transaction type functions to provide more flexibility for handling calls distributed across multiple sites. T-Server translates the input string (ANI string) into a number defined by the translation rules. This processing is called number translation. T-Servers participating in handling calls at multiple sites exchange the translated numbers in order to match the call instances.

The translation process involves two algorithms, one for rule selection and the other for the actual translation. Through the first algorithm, T-Server selects a rule that will be used for number translation. Through the second algorithm, T-Server translates the number according to the selected rule definition. See “Number Translation Rules” on [page 74](#) for more information on configuring rules for your environment.

Number translation occurs as follows:

1. The switch reports a number, typically via `AttributeANI`.
2. T-Server evaluates all configured inbound rules to determine which one is the best fit for the received number. The best fit is determined by comparing the length of, and the specific digits in, the input number with the inbound pattern of each configured rule. See “Rule Examples” on [page 79](#) for specific examples.
3. T-Server translates the number according to the selected rule.

To enable T-Server to translate numbers, you must perform specific configuration tasks that are associated with translation. See “Configuring Number Translation” on [page 81](#).

Number Translation Rules

T-Server uses the number translation rules that you define in the T-Server configuration object in two ways:

- Rule selection—To determine which rule should be used for number translation
- Number translation—To transform the number according to the selected rule

Using ABNF for Rules

The number translation rules must conform to the following syntax, represented using Augmented Backus-Naur Form (ABNF) notation. For more information about ABNF, see RFC 2234, “Augmented BNF for Syntax Specifications: ABNF.”

Note: The following notation explanations begin with the highest level notation. Each explanation includes the name of a component notation and a basic definition of each component that it contains. Some components require more detailed definitions, which are included later in this section.

Common Syntax Notations

Syntax notations common to many of these rules include:

- `*`—Indicates that 0 to an infinite number of the item following this symbol are acceptable.
- `1*`—Indicates that one repetition is required. For T-Server, only one instance is acceptable.
- `/`—Indicates that any of the items mentioned, or a combination of those items, is acceptable.

Component Notations

Component notations include:

- `dialing-plan = *dialing-plan-rule`

where:

- `dialing-plan-rule` represents the name of the rule. Each rule must have a unique name. There are no other naming restrictions, and you do not need to model your names according to the examples in this chapter.

The rules are represented as separate options in the configuration. Also, fields from a rule are represented as parameters in a single option string.

- `rule = [name] in-pattern [out-pattern]`

where:

- `[name]` is the name for the rule option, for example, `rule-01`. In ABNF notation, the brackets `[]` indicate that 0 or 1 instance of the component is required. However, for T-Server, a name is required.
- `in-pattern` is the part of the rule to which T-Server looks when attempting to match the input number.
- `[out-pattern]` is the part of the rule that instructs T-Server on how to translate the input number into the required format. The brackets indicate that either 0 or 1 instance is required. You must create an `out-pattern` for number translation rules.

- `name = *(ALPHA / DIGIT / "-")`

where:

- `ALPHA` indicates that letters can be used in the name for the rule option.
- `DIGIT` indicates that numbers can be used in the name for the rule option.
- `"-"` indicates that a dash (-) can also be used in the option name, for example, `rule-01`.

- `in-pattern = 1*(digit-part / abstract-group)`

where:

- `digit-part` represents numbers. T-Server uses this when selecting the most appropriate rule from the entire dialing plan.
- `abstract-group` represents one or more letters with each letter representing one or more numbers. T-Server uses this when transforming a dial string.

For example, `[1-9]` is the `digit-part` (representing a range of numbers) and `ABBB` is the `abstract-group` for `in-pattern=[1-9]ABBB`.

- `out-pattern = 1*(symbol-part / group-identifier) *param-part`

where:

- `symbol-part` represents digits, symbols, or a combination. Symbols are rarely used. They are not used in the United States.

- `group-identifier` are letters that represent groups of numbers. A letter in the `out-pattern` represents one or more digits, based on the number of times the letter is used in the `in-pattern`.
- `*param-part` represents an additional parameter, such as `phone-context`. Reminder: an asterisk means that 0 to an infinite number of these are acceptable.

For example, in rule-04; `in-pattern=1AAABBBCCC`; `out-pattern=91ABC`, 91 is the `symbol-part`; A, B, and C are `group-identifiers` in the `out-pattern`, each representing three digits, since there are three instances of each in the `in-pattern`.

Note: Prefix an `out-pattern` value with a plus sign (+) for the inbound rule when the output must be in a global form (E.164 format).

- `digit-part = digits / range / sequence`
where:
 - `digits` are numbers 0 through 9.
 - `range` is a series of digits, for example, 1-3.
 - `sequence` is a set of digits.
- `symbol-part = digits / symbols`
where:
 - `digits` are numbers 0 through 9.
 - `symbols` include such characters as +, -, and so on.
- `range = "[" digits "-" digits "]" group-identifier`
where:
 - `"[" digits "-" digits "]"` represents the numeric range, for example, [1-2].
 - `group-identifier` represents the group to which the number range is applied.

For example, [1-2] applies to group identifier A for `in-pattern=[1-2]ABBB`. When T-Server evaluates the rule to determine if it matches the number, it examines whether the first digit of the number, identified as `group-identifier A`, is 1 or 2.

- `sequence = "[" 1*(digits [" , "]) "]" group-identifier`
where:
 - `"[" 1*(digits [" , "]) "]"` represents a sequence of digits, separated by commas, and bracketed. T-Server requires that each digit set have the same number of digits. For example, in [415, 650] the sets have three digits.
 - `group-identifier` represents the group to which the number sequence is applied.

For example, in `in-pattern=1[415,650]A*B`, `[415,650]` applies to group-identifier A. When T-Server evaluates the rule to determine if it matches the number, it examines whether the three digits (group-identifier A) following the 1 in the number are 415 or 650.

- `abstract-group = fixed-length-group / flexible-length-group / entity` where:

- `fixed-length-group` specifies a group composed of a specific number of digits and determined by how many times the group identifier is included in the `in-pattern`. For example, for `in-pattern=1AAABBBCCCC`, there are three digits in group A and B but four in group C.

When you create an `out-pattern`, you include the group identifier only once because the `in-pattern` tells T-Server how many digits belong in that group. For example, `rule-04` (see [page 79](#)) is `in-pattern=1AAABBBCCCC; out-pattern=91ABC`.

- `flexible-length-group` specifies a group composed of 0 or more digits in the group represented by the group-identifier. For example, in `in-pattern=1[415,650]A*B`, `*B` represents the flexible length group containing the remaining digits in the number.
- `entity` represents digits defined for a specific purpose, for example, country code.

The component `abstract-group` is used only for the `in-pattern`.

- `fixed-length-group = 1*group-identifier`

See the earlier explanation under `abstract-group`.

- `flexible-length-group = "*" group-identifier`

See the earlier explanation under `abstract-group`.

- `entity = "#" entity-identifier group-identifier`

where:

- `"#"` indicates the start of a Country Code `entity-identifier`.
- `entity-identifier` must be the letter C which represents Country Code when preceded by a pound symbol (#). Any other letter following the # causes an error.
- `group-identifier` represents the Country Code group when preceded by #C.

The entity component is a special group that assumes some kind of predefined processing, such as the Country Code detection.

- `param-part = ";" param-name "=" param-value`

where:

- `";"` is a required separator element.
- `param-name` is the name of the parameter.
- `"="` is the next required element.
- `param-value` represents the value for `param-name`.

- `param-name = "ext" / "phone-context" / "dn"`
where:
 - "ext" refers to extension.
 - "phone-context" represents the value of the phone-context option configured on the switch.
 - "dn" represents the directory number.
- `param-value = 1*ANYSYMBOL`
where:
 - ANYSYMBOL represents any number, letter, or symbol with no restrictions.
- `group-identifier = ALPHA`
- `entity-identifier = ALPHA`
- `digits = 1*DIGIT`
- `symbols = 1*("-" / "+" / ")" / "(" / ".")`

Recommendations for Rule Configuration

The configuration of rules for inbound numbers usually depends on the settings in the corresponding PBX. These settings often define the form in which the PBX notifies its client applications about the number from which an inbound call is coming.

As a general guideline, configure rules that define how to process calls from:

- Internal numbers.
- External numbers within the same local dialing area.
- External numbers within the same country.
- International numbers.

Rules for inbound numbers, typically for North American locations, might look like this:

1. Two rules to transform internal numbers (extensions):
`name=rule-01; in-pattern=[1-9]ABBB; out-pattern=AB`
`name=rule-02; in-pattern=[1-9]ABBBB; out-pattern=AB`
2. A rule to transform local area code numbers (in 333-1234 format in this example):
`name=rule-03; in-pattern=[1-9]ABBBBBB; out-pattern=+1222AB`
3. A rule to transform U.S. numbers (in +1(222)333-4444 format):
`name=rule-04; in-pattern=1AAAAAAAAA; out-pattern=+1A`
4. A rule to transform U.S. numbers without the +1 prefix (in (222)333-4444 format):
`name=rule-05; in-pattern=[2-9]ABBBBBBBB; out-pattern=+1AB`

5. A rule to transform U.S. numbers with an outside prefix (in 9 +1(222)333-4444 format):
name=rule-06; in-pattern=91AAAAAAAAA; out-pattern=+1A
6. A rule to transform international numbers with an IDD (international dialing digits) prefix (in 011 +44(111)222-3333 format):
name=rule-07; in-pattern=011*A; out-pattern=+A
7. A rule to transform international numbers without an IDD prefix (in +44(111)222-3333 format):
name=rule-08; in-pattern=[2-9]A*B; out-pattern=+AB

Rule Examples

This section provides examples of six rules that are configured as options in the Genesys Configuration Database. It also provides examples of how T-Server applies rules to various input numbers.

Rules

- rule-01** in-pattern=[1-8]ABBB; out-pattern=AB
- rule-02** in-pattern=AAAA; out-pattern=A
- rule-03** in-pattern=1[415, 650]A*B; out-pattern=B
- rule-04** in-pattern=1AAABBBCCCC; out-pattern=91ABC
- rule-05** in-pattern=*A913BBBB; out-pattern=80407913B
- rule-06** in-pattern=011#CA*B; out-pattern=9011AB

Examples

Here are examples of how T-Server applies configured above rules to various input numbers.

Example 1 T-Server receives input number 2326.

As a result of the rule selection process, T-Server determines that the matching rule is rule-01:

```
name=rule-01; in-pattern=[1-8]ABBB; out-pattern=AB
```

The matching count for this rule is 1, because Group A matches the digit 2.

As a result of the parsing process, T-Server detects two groups: Group A = 2 and Group B = 326.

T-Server formats the output string as 2326.

Example 2 T-Server receives input number 9122.

As a result of the rule selection process, T-Server determines that the matching rule is rule-02:

```
name=rule-02; in-pattern=AAAA; out-pattern=A
```

The matching count for this rule is 0; however, the overall length of the input number matches that of the in-pattern configuration.

As a result of the parsing process, T-Server detects one group: Group A = 9122.

T-Server formats the output string as 9122.

Example 3 T-Server receives input number 16503222332.

As a result of the rule selection process, T-Server determines that the matching rule is rule-03:

name=rule-03; in-pattern=1[415, 650]A*B; out-pattern=B

The matching count for this rule is 4, because the first digit matches and all three digits in Group A match.

As a result of the parsing process, T-Server detects two groups: Group A = 650 and Group B = 3222332.

T-Server formats the output string as 3222332.

Example 4 T-Server receives input number 19253227676.

As a result of the rule selection process, T-Server determines that the matching rule is rule-04:

name=rule-04; in-pattern=1AAABBBCCCC; out-pattern=91ABC

The matching count for this rule is 1, because the first digit matches.

As a result of parsing process, T-Server detects three groups: Group A = 925, Group B = 322, and Group C = 7676.

T-Server formats the output string as 919253227676.

Example 5 T-Server receives input number 4089137676.

As a result of rule selection process, T-Server determines that the matching rule is rule-05:

name=rule-05; in-pattern=*A913BBBB; out-pattern=80407913B

The matching count for this rule is 3, because three digits match.

As a result of the parsing process, T-Server detects two groups: Group A = 408 and Group B = 7676.

T-Server formats the output string as 804079137676.

Example 6 T-Server receives input number 011441112223333.

As a result of the rule selection process, T-Server determines that the matching rule is rule-06:

name=rule-06; in-pattern=011#CA*B; out-pattern=9011AB

The matching count for this rule is 3, because three digits match.

As a result of the parsing process, T-Server detects two groups: Group A = 44 and Group B = 1112223333.

T-Server formats the output string as 9011441112223333.

Procedure: Configuring Number Translation

Purpose: To configure the Number Translation feature in T-Server to provide more flexibility for handling calls distributed across multiple sites.

Overview

- The Number Translation feature becomes active when the ISCC/COF feature and/or the `direct-ani` transaction type are used.
- This configuration procedure must be completed within the T-Server Application object corresponding to your T-Server.

Start of procedure

1. Open the T-Server Application's Properties dialog box.
2. Click the Options tab.
3. Create a new section called `extrouter` or open an existing section with this name.
4. Create a new option called `inbound-translator-<n>`. This option points to another section that describes the translation rules for inbound numbers.
5. In this section, create one configuration option for each rule. Specify the rule name as the option name. The values of these options are the rules for the number translation.

For the option description and its valid values, see Chapter 10, “T-Server Common Configuration Options,” on [page 181](#).

6. When you are finished, click Apply.
7. Click OK to save your changes and exit the Properties dialog box.

End of procedure

Network Attended Transfer/Conference Feature

The Network Attended Transfer/Conference (NAT/C) feature is designed to enable agents working in multi-site contact centers to consult with each other before making call transfers or conferences, regardless of whether both agents work at the same or different sites. It also enables the agent who requests a consultation to maintain his or her conversation with the customer while the system is looking for an available agent and setting up the consultation call.

The NAT/C feature does not rely on the call transfer capabilities of the local switch.

There are two modes in which the network attended transfer/conference can be performed: *direct* and *URS-controlled*. Figure 12 shows the sequence of steps that occur in *URS-controlled* mode, when Agent A, who is handling a customer call, requests a consultation with another agent, and URS (Universal Routing Server) selects Agent B, who is working at another site. The *direct* mode is similar to the *URS-controlled* mode, with the difference that URS is not involved in the process (Step 2 and Step 3 are omitted).

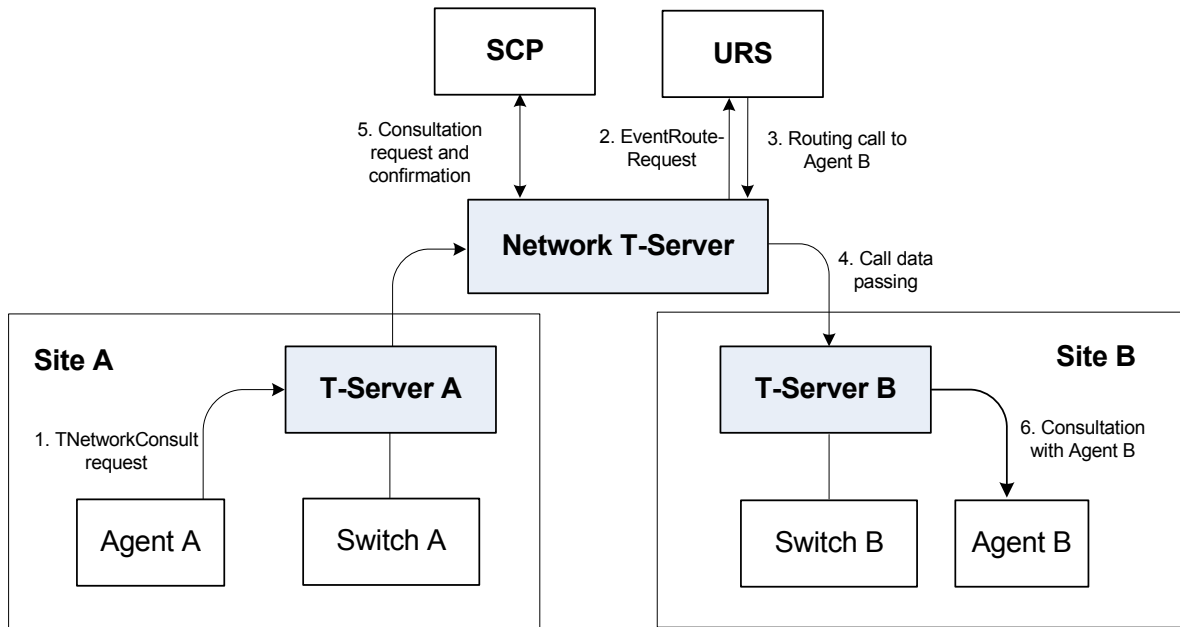


Figure 12: Steps in the NAT/C Process in URS-Controlled Mode

Step 1

Agent A makes a request for a consultation with another agent. A `TNetworkConsult` request is relayed to the Network T-Server. Depending on the parameter settings of the `TNetworkConsult` request, the NAT/C feature will operate in either *direct* or *URS-controlled* mode. For more information, see the *Voice Platform SDK 8.x .NET (or Java) API Reference*.

Step 2

(*URS-controlled* mode only.) The Network T-Server sends `EventRouteRequest` to URS.

Step 3

(*URS-controlled* mode only.) URS locates an available agent at Site B and instructs the Network T-Server to route the call to Agent B. The Network

T-Server confirms the initiation of the network transfer by sending `EventNetworkCallStatus` to T-Server A, which then relays it to Agent A.

Step 4

The Network T-Server proceeds to obtain the access number from T-Server B, and passes the call data to T-Server B. (See “ISCC Call Data Transfer Service” on [page 49](#) for details.)

Step 5

The Network T-Server instructs the Service Control Point (SCP) to initiate a new voice path with Agent B. Once the connection is confirmed, the Network T-Server distributes `EventNetworkCallStatus` to both T-Server A and T-Server B, which then relay it to Agent A and Agent B respectively, to indicate that the consultation call is being established.

The Network T-Server also distributes `EventRouteUsed` to URS to confirm successful routing of the call to the selected agent.

Step 6

At this point, the customer is on hold, and Agent A is consulting with Agent B. Agent A can do one of the following:

- End the consultation and retrieve the original customer call
- Alternate between Agent B and the customer
- Set up a conference call with Agent B and the customer
- Transfer the customer call to Agent B

Note: All T-Servers support NAT/C requests with `AttributeHomeLocation` provided that this attribute identifies a network location that is capable of processing such requests. Refer to the *Network T-Server Deployment Guides* to determine whether a specific Network T-Server can process these requests.

Event Propagation Feature

The Event Propagation feature complements the ISCC and ISCC/COF features by distributing updated user data and party-related events to remote T-Servers. This feature is used when a call is being made, transferred, or conferenced to another location, and when, as a result, one or more instances of the call reside at one location while other call instances reside at another location. In this scenario, when a client at one location makes changes to user data, updated user data is passed (*propagated*) to T-Servers at other locations.

The Event Propagation feature consists of User Data update propagation and Party Events propagation.

User Data Propagation

User data propagation takes place when a client at one location makes changes to user data associated with a call that was made, transferred, conferenced, or routed to other locations. The remote clients involved with the call are notified about the changes with `EventAttachedDataChanged`.

When T-Server receives a local update to user data (that is, when a client of this T-Server has changed the call's user data), T-Server determines if parties at remote locations are involved with the call and, if so, sends (propagates) the updated user data to the T-Servers at remote locations.

When T-Server receives a remote update to user data (that is, when a client of a remote T-Server has changed the call's user data and the remote T-Server has used the Event Propagation feature to send the updated user data), T-Server:

1. Updates the user data of the corresponding local call.
2. Determines if parties at other remote locations are involved with the call and, if so, propagates the updated user data to T-Servers at other remote locations.

The locations to which user data is propagated are selected based on a call distribution topology. That is, the updated user data is passed directly to the location to which a call was sent and to the location from which the call was received, excluding the location from which the update was received.

For example, consider a call made from location A to location B, and then conferenced from location B to location C. The three instances of the call reside at different locations: the first instance is at location A, the second instance is at location B, and the third instance is at location C. The Event Propagation feature is employed in the following scenarios:

- When T-Server at location A receives a local update to user data, it notifies T-Server at location B (to which it sent the call) about changes to the call's user data. Thus, T-Server at location B receives a remote update to user data and, in turn, notifies T-Server at location C (to which it sent the call) about these changes.

Although T-Server at location C receives a remote update to user data, it does not pass the notification to any other T-Servers, because it did not send the call to any other locations. As mentioned earlier, T-Servers at locations B and C update the user data of the corresponding local calls and notify their clients about the changes with `EventAttachedDataChanged`.

- When T-Server at location B receives a local update to user data, it notifies T-Server at location C (to which it sent the call) and T-Server at location A (from which it received the call) about changes to the call's user data. Thus, T-Servers at locations C and A receive a remote update to user data.

Because T-Server at location C did not send the call to any other locations, and T-Server at location A originated the call, neither of these T-Servers passes the notification to any other T-Servers. T-Servers at locations C and A update the user data of the corresponding local calls and notify their clients about the changes with `EventAttachedDataChanged`.

- When T-Server at location C receives a local update to user data, it notifies T-Server at location B (from which it received the call) about changes to the call's user data. Thus, T-Server at location B receives a remote update to user data and, in turn, notifies T-Server at location A (from which it received the call) about these changes.

Although T-Server at location A receives a remote update to user data, it does not pass the notification to any other T-Servers, because it originated the call. T-Servers at locations B and A update the user data of the corresponding local calls and notify their clients about the changes with `EventAttachedDataChanged`.

When a call is distributed between location A and location C using location B, and is then deleted on location B, propagation between locations A and C still occurs through the transit node at location B.

Party Events Propagation

Party events propagation takes place when a transfer or a conference is completed for a call that was made to or from one or more remote locations, or when a conference party is removed from the conference.

In these cases, the Event Propagation feature distributes party events, such as `EventPartyChanged`, `EventPartyAdded`, and `EventPartyDeleted`, to remote locations involved with the call, according to appropriate call model scenarios.

For example, consider a call made from DN 1 to DN 2 on location A. A `TInitiateConference` request is then issued for DN 2 to transfer the call to external DN 3 on location B. That transfer is made by means of ISCC routing. When this conference is completed on location A, the Event Propagation feature sends `EventPartyChanged` to location B and distributes this event to involved client applications that are connected to location B and registered for DN 3. After that, if a party of the conference is removed from the conference (for example, a party on DN 2), the Event Propagation feature sends `EventPartyDeleted` to location B and distributes this event to client applications registered for DN 3.

If a call involved in the propagation has no local parties but has two or more remote parties, the party events propagation is processed in the same manner as the propagation of user data updates.

For a complete event flow in such scenarios, refer to the *Genesys Events and Models Reference Manual*.

Switch Partitioning

A multi-site environment with switch partitioning or intelligent trunks can be defined as a configuration of multiple virtual switches (or Switch objects) that are defined in Configuration Manager under a single Switching Office object representing a physical switch. Each Switch object has its own instance of a T-Server application. All T-Server applications connect to the switch via the same or different CTI link or a gateway. (See [Figure 13](#).)

When the Event Propagation feature is active, updated user data and party-related events—`EventPartyChanged`, `EventPartyDeleted`, and `EventPartyAdded`—are propagated to T-Servers that are involved in call transactions, such as transfer or conference. However, with switch partitioning, the call instances may reside at one partition or at different partitions.

Site A

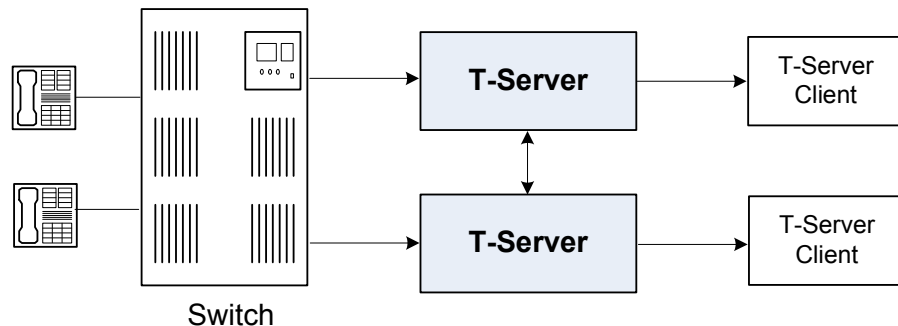


Figure 13: Switch Partitioning Architecture

Starting with version 8.0, in addition to `ConnIDs` and `UserData`, T-Server can synchronize the `CallType` attribute. Each T-Server is required to register all DNs it monitors. In a multi-partitioned environment, when configured, calls between partitions are reported as internal (`CallTypeInternal`). In a non-partitioned environment, such calls are reported as inbound (`CallTypeInbound`) and/or outbound (`CallTypeOutbound`), depending on the direction of a call. In order for T-Servers to report calls between specified partitions as internal, registered DNs of these partitions must be assigned to a Switch (T-Server), Switching Office, or Tenant, using the [dn-scope](#) configuration option. If DNs that are involved in calls are not in the T-Server scope, those DNs will be reported as inbound or outbound.

In addition, T-Server supports `LocalCallType` and `PropagatedCallType` attributes, which depend on the [propagated-call-type](#) configuration option setting for reporting. See the option description on [page 186](#).

To control race conditions that may occur in the switch-partitioned environment, use the `epp-tout` configuration option (see [page 201](#)).

Notes: Because of possible delays in TCP/IP connections, a sequence of events sent for the same call by two or more T-Servers to clients may appear in an unexpected order. For example, in a simple call scenario with two partitions, `EventRinging` and `EventEstablished` messages may both arrive before `EventDialing`.

Genesys switch partitioning does not apply to hardware partitioning functionality that is supported on some switches.

[Table 5](#) shows the T-Server types that support switch partitioning.

Table 5: T-Server Support for Switch Partitioning

T-Server Type	Supported
Alcatel A4400/OXE	Yes
Avaya Communication Manager	Yes
Avaya TSAPI	Yes
Cisco Unified Communications Manager	Yes
SIP Server	Yes

Event Propagation Configuration

The basic Event Propagation feature configuration includes a setting of specific configuration options at a T-Server Application level. The advanced feature configuration allows you to customize the feature at a Switch level.

When determining whether to notify other T-Servers of changes to user data, or to distribute party events, T-Server checks:

1. Call topology (what location a call came from and to what location the call was then transferred or conferenced).
2. Outbound parameters of the Switch this T-Server relates to (whether propagation parameters are configured for the access codes this switch uses to reach the switch at the location a call came from and the switch at the location to which the call was then transferred or conferenced).

Warning! The direction of user-data or party-events propagation does not necessarily match the direction of call distribution. Therefore, the access code used to deliver the call can differ from the access code used for the purpose of Event Propagation.

If one of the T-Servers along the call distribution path has the Event Propagation feature disabled, that T-Server does not distribute events to remote locations.

Procedure:**Activating Event Propagation: basic configuration**

Purpose: To activate the Event Propagation feature for User Data updates and call-party-associated events (Party Events) distribution.

Start of procedure

1. Open the T-Server Application's Properties dialog box.
2. Click the Options tab.
3. Open the extrouter section.
4. Set the [event-propagation](#) option to the list value.
This setting enables User Data propagation. If you need to enable Party Events propagation, perform Step 5.
5. Set the [use-data-from](#) option to the current value.
This setting enables Party Events propagation.
For the option description and its valid values, see Chapter 10, “T-Server Common Configuration Options,” on [page 181](#).
6. When you are finished, click Apply.
7. Click OK to save your changes and exit the Properties dialog box.

End of procedure**Next Steps**

- For advanced feature configuration, do the following procedure:
[Procedure: Modifying Event Propagation: advanced configuration](#), on [page 88](#)

Procedure:**Modifying Event Propagation: advanced configuration**

Purpose: To modify access codes for advanced Event Propagation configuration.

Prerequisites

- [Procedure: Activating Event Propagation: basic configuration](#), on [page 88](#)

Overview

You can set Event Propagation parameters using:

- The Default Access Code properties of the Switch that receives an ISCC-routed call (the destination switch).
- The Access Code properties of the Switch that passes an ISCC-routed call (the origination switch).

If you do not set up Event Propagation parameters for a given Access Code, T-Server uses corresponding settings configured for the Default Access Code of the destination switch.

The procedures for modifying Default Access Codes and Access Codes are very similar to each other.

Start of procedure

1. Among configured Switches, select the Switch that the configured T-Server relates to.
2. Open the Switch's Properties dialog box and click either the Default Access Codes tab or the Access Codes tab.
3. Select a configured Default Access Code or configured Access Code and click Edit.

Note: If no Default Access Code is configured, see [page 93](#) for instructions. If no Access Codes are configured, see [page 94](#) for instructions.

4. In the Switch Access Code Properties dialog box that opens, specify a value for the ISCC Protocol Parameters field as follows:
 - To enable distribution of both user data associated with the call and call-party-associated events¹, type:
propagate=yes
which is the default value.
 - To enable distribution of user data associated with the call and disable distribution of call-party-associated events, type:
propagate=udata
 - To disable distribution of user data associated with the call and enable distribution of call-party-associated events, type:

-
1. The following are call-party-associated events: EventPartyChanged, EventPartyDeleted, and EventPartyAdded.

- propagate=party
 - To disable distribution of both user data associated with the call and call-party-associated events, type:
propagate=no
- 5. Click OK to save configuration updates and close the Switch Access Code Properties dialog box.
- 6. Click Apply and OK to save configuration updates and close the Switch Properties dialog box.

End of procedure

ISCC Transaction Monitoring Feature

This feature allows T-Server clients to monitor ISCC transactions that occur during the call data transfer between T-Servers in a multi-site environment.

In order to be able to monitor ISCC messaging, a T-Server client must subscribe to the ISCC Transaction Monitoring. Once a subscription request is confirmed, a client will receive updates about all multi-site operations of this T-Server.

The `TTransactionMonitoring` request is used to instruct T-Server to start, stop, or modify a client's subscription to Transaction Monitoring feature notifications by setting the `TSubscriptionOperationType` parameter to `SubscriptionStart`, `SubscriptionStop`, or `SubscriptionModify` respectively. The transaction status is reported in `EventTransactionStatus` messages to the subscribed clients.

To determine whether the Transaction Monitoring feature is supported by a specific T-Server, a T-Server client may query T-Server's capabilities. For more information about support of this feature, see *Genesys Events and Models Reference Manual* and *Voice Platform SDK 8.x .NET (or Java) API Reference*.

Configuring Multi-Site Support

Prior to configuring T-Server to support multi-site operation, you must read the previous sections of this chapter on multi-site deployment. In particular, Table 3 on [page 65](#) shows which transaction types are supported by a specific T-Server, while Table 4 on [page 70](#) shows whether your T-Server supports the

NetworkCallID attribute for the ISCC/COF feature. Use this information as you follow the instructions in this chapter.

Note: Before attempting to configure a multi-site environment, Genesys recommends that you plan the changes you want to make to your existing contact centers. You should then gather the configuration information you will need (such as the name of each T-Server application, port assignments, and switch names), and use Configuration Manager to create and partially configure each T-Server object. Review multi-site option values in the “extrouter Section” on [page 191](#) and determine what these values need to be, based on your network topology.

For T-Server to support multi-site operation, you must create and configure three types of objects in the Configuration Layer:

1. Applications
2. Switches, including Access Codes
3. DNs

You must configure these objects for origination and destination locations. Multi-site support features activate automatically at T-Server startup. See “DNs” on [page 98](#) for details.

Applications

Ensure that T-Server Application objects, and their corresponding Host objects, exist and are configured for origination and destination locations.

Once you’ve done that, use Configuration Manager to add this configuration to a T-Server Application.

Procedure: Configuring T-Server Applications

Purpose: To configure T-Server Application objects for multi-site operation support.

Start of procedure

1. Open the T-Server Application's Properties dialog box.
2. Click the Connections tab and click Add to add a connection to the appropriate T-Server. The Connection Info Properties dialog box displays.
3. Use the Browse button to search for the T-Server you want to connect to, and fill in the following values:
 - Port ID

- Connection Protocol
 - Local Timeout
 - Remote Timeout
 - Trace Mode
4. Click the Options tab. Create a new section called extrouter or open an existing section with this name.

Note: If you do not create the extrouter section, T-Server uses the default values of the corresponding configuration options.

5. Open the extrouter section. Configure the options used for multi-site support.

Note: For a list of options and valid values, see “extrouter Section” on [page 191](#), in the “T-Server Common Configuration Options” chapter in Part Two of this document.

6. When you are finished, click Apply.
7. Repeat this procedure for all T-Servers for origination and destination locations that are used for multi-site operations.

End of procedure

Next Steps

- See “[Switches and Access Codes.](#)”

Switches and Access Codes

Ensure that Switching Office and Switch objects are configured for both origination and destination locations.

You configure Access Codes to a destination switch in the origination Switch's Properties dialog box. The only exception is the Default Access Code, which is configured at the destination Switch's Properties dialog box.

You can configure two types of switch Access Codes in the Switch's Properties dialog box:

- A Default Access Code (for inbound calls)—Specifies the access code that other switches can use to access this switch when they originate a multi-site transaction.
- An Access Code (for outbound calls)—Specifies the access code that this switch can use when it originates a multi-site transaction to access another switch.

When the origination T-Server processes a multi-site transaction, it looks for an access code to the destination switch. First, T-Server checks the Access Code of the origination Switch:

- If an access code to the destination switch is configured with the target type Target ISCC and with any transaction type except Forbidden, T-Server uses this access code to dial the destination switch.
- If the access code to the destination switch is not configured on the Access Code tab of the origination switch, the origination T-Server checks the Default Access Code tab of the destination switch. If an access code is configured there with the target type Target ISCC and with any transaction type except Forbidden, T-Server uses this access code to dial the destination switch.
- If no access code with the required properties is found, T-Server rejects the transaction.

Note: When migrating from previous releases of T-Servers to 8.1, or when using T-Servers of different releases (including 8.1) in the same environment, see “Compatibility Notes” on [page 97](#).

Procedure: Configuring Default Access Codes

Purpose: To configure the Default Access Codes (one per Switch object) to be used by other switches to access this switch when they originate a multi-site transaction.

Prerequisites

- Ensure that Switching Office and Switch objects are configured for both origination and destination locations.

Start of procedure

1. Among configured Switches, select the Switch that the configured T-Server relates to.
2. Open the Switch Properties dialog box and click the Default Access Codes tab.
3. Click Add to open the Access Code Properties dialog box.

4. In the **Code** field, specify the access code used by remote switches to reach a DN at this switch. An access code is used as a prefix to the remote switch numbers.

Note: If no prefix is needed to dial to the configured switch, you can leave the **Code** field blank.

5. In the **Target Type** field, select **Target ISCC**.
6. In the **Route Type** field, select a value corresponding to the transaction type you want to use (given that it is supported for your switch type).
7. When you are finished, click **Apply**.

End of procedure

Next Steps

- See [“Configuring Access Codes.”](#)

Procedure: Configuring Access Codes

Purpose: To configure the **Access Codes** (one or more per **Switch** object) that this switch can use when it originates a multi-site transaction to access another switch.

Prerequisites

- Ensure that **Switching Office** and **Switch** objects are configured for both origination and destination locations.

Start of procedure

1. Among configured **Switches**, select the **Switch** that the configured **T-Server** relates to.
2. Open the **Switch Properties** dialog box and click the **Access Codes** tab.
3. Click **Add** to open the **Access Code Properties** dialog box.
4. In the **Switch** field, specify the switch that this switch can reach using this access code. Use the **Browse** button to locate the remote switch.

5. In the Code field, specify the access code used to reach a DN at the remote switch from this switch. An access code is used as a prefix to the remote switch numbers.

Note: If no prefix is needed to dial from one switch to another, you can leave the Code field blank.

6. In the Target Type field, select Target ISCC.

When you select Target ISCC as your target type, the Properties dialog box changes its lower pane to the Sources pane. It is here that you enter the extended parameters for your access codes, by specifying the ISCC Protocol and ISCC Call Overflow Parameters.

To set these parameters, locate the two drop-down boxes that appear below the Target Type field in the Sources pane of that Properties dialog box.

- a. In the ISCC Protocol Parameters drop-down box, enter the appropriate ISCC Protocol parameter, as a comma-separated list of one or more of the following items shown in [Table 6](#):

Table 6: Target Type: ISCC Protocol Parameters

ISCC Protocol Parameters	Description
dnis-tail=<number-of-digits>	Where number-of-digits is the number of significant DNIS digits (last digits) used for call matching. 0 (zero) matches all digits.
propagate=<yes, udata, party, no>	Default is yes. For more information, see “Modifying Event Propagation: advanced configuration” on page 88 .
direct-network-callid=<>	For configuration information, see Part Two of this document. (Use Table 4 on page 70 to determine if your T-Server supports the direct-network-callid transaction type.)

- b. In the ISCC Call Overflow Parameters drop-down box, enter call overflow parameters, as a comma-separated list of one or more of the following items shown in [Table 7](#):

Table 7: Target Type: ISCC Call Overflow Parameters

ISCC Call Overflow Parameters	Description
match-callid	Matches calls using network CallID.
match-ani	Matches calls using ANI. Note: When using match-ani, the match-flexible parameter must be set to false.
match-flexible	Supports flexible call matching based on the following values: Default Value: true Valid Values: true, false, and [matching-context-type], where [matching-context-type] is the switch-specific value, which must be the same as the value of the default-network-call-id-matching configuration option of the corresponding T-Server.
inbound-only=<boolean>	Default is true. Setting inbound-only to true disables COF on consultation and outbound calls.

7. In the Route Type field, select a value corresponding to the transaction type you want to use (given that it is supported for your switch type). [Table 8](#) contains cross-reference information on transaction types that the Configuration Layer and T-Server use.

Table 8: Route Type and ISCC Transaction Type Cross-Reference

Route Type Field Value	ISCC Transaction Type
Default	The first value from the list of values specified in the cast-type option for the T-Server at the destination site
Direct	direct-callid
Direct ANI	direct-ani
Direct Digits	direct-digits
Direct DNIS and ANI	Reserved

Table 8: Route Type and ISCC Transaction Type Cross-Reference (Continued)

Route Type Field Value	ISCC Transaction Type
Direct Network Call ID	direct-network-callid
Direct No Token	direct-notoken
Direct UUI	direct-uui
DNIS Pooling	dnis-pooling
Forbidden	External routing to this destination is not allowed
ISCC defined protocol	Reserved
PullBack	pullback
Re-Route	reroute
Route	route

8. When you are finished, click Apply.

End of procedure

Next Steps

- After configuring a switch for multi-site support, proceed with the configuration of DN's assigned to this switch.

Compatibility Notes

When migrating from previous releases of T-Servers to 8.1, or when using T-Servers of different releases (including 8.1) in the same environment, keep in mind the following compatibility issues:

- The Target External Routing Point value of the Target Type field is obsolete and provided only for backward compatibility with T-Servers of releases 5.1 and 6.0. When two access codes for the same switch are configured, one with the Target ISCC target type and the other with the Target External Routing Point target type, T-Servers of releases 8.x, 7.x, 6.5, and 6.1:
 - Use the Target ISCC access code for transactions with T-Servers of releases 8.x, 7.x, 6.5, and 6.1.
 - Use the Target External Routing Point access code for transactions with T-Servers of releases 5.1 and 6.0.

When the only access code configured for a switch has the Target External Routing Point target type, T-Server uses this access code for all transactions.

- When the Target External Routing Point value of the Target Type field is configured, you must set the Route Type field to one of the following:
 - Default to enable the route transaction type
 - Label to enable the direct-ani transaction type
 - Direct to enable the direct transaction type

Note: The direct transaction type in releases 5.1 and 6.0 corresponds to the direct-callid transaction type in releases 6.1 and later.

- UseExtProtocol to enable the direct-uuu transaction type
- PostFeature to enable the reroute transaction type

These values are fully compatible with the transaction types supported in T-Server release 5.1.

- For successful multi-site operations between any two locations served by release 5.1 T-Servers, identical Route Type values must be set in the Switch's Access Code Properties dialog boxes for both the origination and destination switches.

DNs

Use the procedures from this section to configure access resources for various transaction types.

Procedure: Configuring access resources for the route transaction type

Purpose: To configure dedicated DNs required for the route transaction type.

Prerequisites

- Ensure that Switching Office and Switch objects are configured for both origination and destination locations.

Start of procedure

1. Under a configured Switch, select the DNs folder. From the main menu, select **File > New > DN** to create a new DN object.
2. On the **General** tab of the DN's **Properties** dialog box, specify the number of the configured DN as the value of the **Number** field. This value must correspond to the Routing Point number on the switch.
3. Select **External Routing Point** as the value of the **Type** field.
4. If a dialable number for that Routing Point is different from its DN name, specify the number in the **Association** field.
5. Click the **Access Numbers** tab. Click **Add** and specify these access number parameters:
 - Origination switch.
 - Access number that must be dialed to reach this DN from the origination switch.

In determining an access number for the Routing Point, T-Server composes it of the values of the following properties (in the order listed):

- a. Access number (if specified).
- b. Switch access code from the switch of the origination party to the switch to which the Routing Point belongs, concatenated with its **Association** (if the **Association** value is specified).
- c. Switch access code from the switch of the origination party to the switch to which the Routing Point belongs, concatenated with the number for the DN.
- d. Default access code of the switch to which the Routing Point belongs, concatenated with its **Association** (if the **Association** value is specified).
- e. Default access code of the switch to which the Routing Point belongs, concatenated with the number for the DN.

Note: If option `use-implicit-access-numbers` is set to true, the access number composed of switch access code and DN can be used for external transfers of calls originating at switches for which an access number is not specified.

6. When you are finished, click **Apply**.

End of procedure

Procedure:

Configuring access resources for the dnis-pool transaction type

Purpose: To configure dedicated DN's required for the dnis-pool transaction type.

Start of procedure

1. Under a configured Switch, select the DN's folder. From the main menu, select **File > New > DN** to create a new DN object.
2. On the **General** tab of the DN's **Properties** dialog box, specify the number of the configured DN as the value of the **Number** field. This value must be a dialable number on the switch.
3. Select **Access Resource** as the **Type** field and type **dnis** as the value of the **Resource Type** field on the **Advanced** tab.
4. Click the **Access Numbers** tab. Click **Add** and specify these **Access Number** parameters:
 - Origination switch.
 - Access number that must be dialed to reach this DN from the origination switch.

An access number for the access resource is determined in the same manner as for the route access resource.

5. When you are finished, click **Apply**.

End of procedure

Procedure:

Configuring access resources for direct-* transaction types

Start of procedure

You can use any configured DN as an access resource for the **direct-*** transaction types. (The * symbol stands for any of the following: **callid**, **uvi**, **notoken**, **ani**, or **digits**.)

You can select the **Use Override** check box on the **Advanced** tab to indicate whether the override value should be used instead of the number value to dial to the DN. You must specify this value if the DN has a different DN name and dialable number. In fact, this value is required for T-Servers for some switch

types—such as Aspect ACD, Nortel Communication Server 2000/2100, and Spectrum.

End of procedure

Procedure: Configuring access resources for ISCC/COF

Purpose: To configure dedicated DNs required for the ISCC/COF feature.

Start of procedure

Note: Use Table 4 on [page 70](#) to determine if your T-Server supports the ISCC/COF feature.

1. Under a configured Switch, select the DNs folder. From the main menu, select **File > New > DN** to create a new DN object.
2. On the **General** tab of the **DN Properties** dialog box, enter the name of the configured DN in the **Number** field.

Note: The name of a DN of type **Access Resource** must match the name of a DN in your configuration environment (typically, a DN of type **Routing Point** or **ACD Queue**), so T-Server can determine whether the calls arriving at this DN are overflowed calls.

3. Select **Access Resource** as the value for the **Type** field.
4. On the **Advanced** tab, type **cof-in** or **cof-not-in** as the value for the **Resource Type** field.

Note: Calls coming to DNs with the **cof-not-in** value for the **Resource Type** are never considered to be overflowed.

5. When you are finished, click **Apply**.

End of procedure

Procedure: Configuring access resources for non-unique ANI

Purpose: To configure dedicated DNs required for the non-unique-ani resource type.

The `non-unique-ani` resource type is used to block `direct-ani` and `COF/ani` from relaying on ANI when it matches configured/enabled resource digits. Using `non-unique-ani`, T-Server checks every ANI against a list of `non-unique-ani` resources.

Start of procedure

1. Under a configured Switch, select the DNs folder. From the main menu, select **File > New > DN** to create a new DN object.
2. On the **General** tab of the **DN Properties** dialog box, specify the ANI digits that need to be excluded from normal processing.
3. Select **Access Resource** as the value for the **Type** field.
4. On the **Advanced** tab, specify the **Resource Type** field as `non-unique-ani`.
5. When you are finished, click **Apply**.

End of procedure

Procedure: Modifying DNs for isolated switch partitioning

Purpose: To modify DNs that belong to a particular partition where switch partitioning is used.

This configuration instructs T-Server to select an External Routing Point that has the same partition as the requested destination DN.

Note: When a target DN is not configured or has no configured partition name, T-Server allocates a DN of the **External Routing Point** type that belongs to any partition.

Start of procedure

1. Under a Switch object, select the DNs folder.
2. Open the **Properties** dialog box of a particular DN.
3. Click the **Annex** tab.
4. Create a new section named **TServer**.
5. Within that section, create a new option named **epn**. Set the option value to the partition name to which the DN belongs.
6. Repeat Steps 1–5 for all DNs, including DNs of the **External Routing Point** type, that belong to the same switch partition.

7. When you are finished, click Apply.

End of procedure

Configuration Examples

This section provides two configuration examples and describes how the configuration settings affect T-Server's behavior.

Multiple Transaction Types

This example demonstrates the difference in how ISCC directs a call when you specify two different transaction types (`route` and `direct-ani`).

In this example, you configure an origination and a destination switch for as described in “Switches and Access Codes” on [page 92](#).

1. Among configured Switches, select the origination Switch.
2. Open the Switch Properties dialog box and click the Default Access Codes tab.
3. Click Add to open the Access Code Properties dialog box.
4. Set the Access Code field to 9.
5. When you are finished, click Apply.
6. Among configured Switches, select the destination Switch.
7. Under the destination Switch, configure a DN as described in “Configuring access resources for the route transaction type” on [page 98](#).
8. Set the DN Number field to 5001234567.
9. Click the Advanced tab of this DN's Properties dialog box.
10. Select the Use Override check box and enter 1234567 in the Use Override field.
11. When you are finished, click Apply or Save.
12. Use a T-Server client application to register for this new DN with the destination T-Server and, therefore, with the switch.
13. Request to route a call from any DN at the origination switch to the destination DN you have just configured:
 - If you are using the `route` ISCC transaction type, the client requests that T-Server deliver a call to a destination location using the DN number 5001234567. ISCC requests that the switch dial one of the external routing points at the destination location, using the value either of the Access Number field or of the Access Code field, which is 9, concatenated with the external routing point at the destination location. The call is routed to the DN number 5001234567.

- If you are using the `direct-ani` ISCC transaction type, the client requests that T-Server deliver a call to a destination location using the DN number 1234567, which is the `Use Override` value. ISCC requests that the switch dial 91234567, which is a combination of the `Switch Access Code` value and the `Use Override` value. The destination T-Server is waiting for the call to directly arrive at DN number 5001234567.

Call Overflow Methods

This section demonstrates how to indicate which overflow methods a switch supports.

In this example, for T-Server to use ANI/OtherDN matching in call overflow and manual transfer scenarios, set the ISCC Call Overflow Parameters to:

```
match-ani, inbound-only=true
```

when configuring Switch Access Codes as described on [page 94](#).

With this setting, the switch's location is queried for call data each time the destination T-Server receives an inbound call with the ANI or OtherDN attribute.

For T-Server to use NetworkCallID matching in call overflow and manual transfer scenarios, set the ISCC Call Overflow Parameters to (for example):

```
match-callid, inbound-only=false
```

when configuring Switch Access Codes as described on [page 94](#).

With this setting, the switch's location is queried for call data each time the destination T-Server receives a call of any type (including inbound) with the NetworkCallID attribute.

4

Pre-Installation Setup

This chapter describes the setup tasks that you must complete before you can install the IVR Server or an IVR Driver.

Note: If you use the IVR Interface Option Wizard that is available with IVR Interface Option 8.1, most of these pre-installation tasks described in this chapter are unnecessary. Although you must still manually install prerequisite products before you install IVR Server or an IVR Driver, you can use the wizard to import application templates, and to create the Application objects for IVR Server 8 and IVR Driver 8.

This chapter contains the following sections:

- [IVR Interface Option Application Templates, page 106](#)
- [Setting up the IVR_Driver Application, page 106](#)
- [Setting up the I-Server Application, page 110](#)
- [Setting up the TServer_IVR Application, page 114](#)
- [Setting up the TServer_IVR_Network Application, page 119](#)

For information about how to install an IVR Driver, see the *IVR Interface Option 8.1 System Administrator's Guide* for your particular IVR Driver.

Note: The examples in this chapter are based on IVR Server 8.1. The release number of the actual application templates that you import into your system might differ, depending on the current release of your IVR Server.

IVR Interface Option Application Templates

[Table 9](#) lists the application templates that you must import and configure in Configuration Manager, depending on your configuration mode.

Table 9: Application Templates and Configuration Modes

Application Template	Configuration Modes Required	Notes
IVR_Driver	IVR-In-Front IVR-Behind-Switch	A component of the IVR Driver product, this application template enables communication between the IVR Driver and the IVR Server. This template is available on the IVR Driver CD. Note: The IVR Driver 8.0 for CONVERSANT does not require, and cannot use, the IVR_Driver application template.
I-Server	IVR-In-Front IVR-Behind-Switch	A component of the IVR Server product, this application template enables Load Balancing, statistics, and other IVR Server functions. This template is available on the IVR Server CD.
TServer_IVR	IVR-In-Front IVR-Behind-Switch	A component of the IVR Server product, this application template starts and stops the IVR Server, enables network logging, and communicates with other Genesys products (for example, Configuration Server). This template is available on the IVR Server CD.
TServer_IVR_Network	IVR Network T-Server	This application template provides similar functions to the TServer_IVR application template. It includes the capability to communicate with a Network T-Server. This template is available on the IVR Server CD.

Setting up the IVR_Driver Application

This section describes how to set up the IVR_Driver application. It contains the following subsections:

- “Importing the IVR_Driver application template” on [page 107](#)
- “Defining the IVR_Driver application” on [page 107](#)
- “Configuring the IVR_Driver Application” on [page 108](#)

Procedure: Importing the IVR_Driver application template

Purpose: To import the IVR_Driver application template into Configuration Manager.

Start of procedure

1. In Configuration Manager, expand Environment, right-click Application Templates, and select Import Application Template.
2. Navigate to the directory that contains the IVR_Driver template file (IVRDriver_810.apd), select the file, and click Open. The New Application Template Properties dialog box appears.
3. In the Name box, accept the default value, or enter a new name for the IVR_Driver template.
4. In the Type box, verify that IVR Driver is selected.
5. In the Version box, verify that the correct version number is selected.
6. Click OK to import the IVR_Driver application template.

End of procedure

Next Steps

You are now ready to define the IVR_Driver application.

Procedure: Defining the IVR_Driver application

Purpose: To define the IVR_Driver application in Configuration Manager.

Start of procedure

1. Expand Environment, right-click Applications, and select New > Application. A list of the available application templates is displayed.
2. Select the IVR_Driver application template that you defined in “Importing the IVR_Driver application template” on [page 107](#), and then click OK. The New IVRDriver_810 Properties dialog box appears, with the General tab displayed.
3. In the Name box, accept the default value, or enter a new name for the IVR_Driver application.

4. Click OK to create the IVR_Driver application and close the New IVRDriver_810 Properties dialog box; or click Apply to create the application without closing the dialog box, so that you can perform additional actions.

End of procedure

Next Steps

You are now ready to configure the IVR_Driver application.

Configuring the IVR_Driver Application

The following sections describe how to configure the IVR_Driver application in Configuration Manager:

- “Assigning an IVR_Driver host” on [page 108](#)
- “Defining start parameters for the IVR_Driver application” on [page 109](#)
- “Adding connections for the IVR_Driver application” on [page 109](#)

Procedure: Assigning an IVR_Driver host

Prerequisites

The vendor-provided IVR must be configured on the same host.

Purpose: To assign an IVR_Driver host.

Start of procedure

1. In the IVRDriver_810 Properties dialog box, click the Server Info tab.
2. In the Host box, select the host on which the IVR_Driver application is installed.

Note: In order for a host to appear for selection in the drop-down list, you must predefine it by doing one of the following:

- Clicking the Folder icon next to the Host box.
 - In the main Configuration Manager window, select Environment > Host.
-

3. In the Ports box, add a new port number or edit an existing one.
4. In the Backup Server box, accept the default value ([None]), because this option is not supported for the IVR_Driver application.

5. In the Reconnect Timeout and Reconnect Attempts boxes, accept the default values.
6. Click Apply to assign the IVR_Driver host.

End of procedure

Procedure: Defining start parameters for the IVR_Driver application

Purpose: To define the start parameters for the IVR_Driver application.

Start of procedure

1. In the IVRDriver_810 Properties dialog box, click the Start Info tab.
2. Enter a single period (.) in the following boxes:
 - Working Directory
 - Command Line
 - Command Line Arguments

Note: These three boxes will be updated during the installation process.

3. In the Startup and Shutdown boxes, accept the default values.
4. Make sure that the Auto-Restart check box is not selected.
5. Click Apply to define the start parameters.

End of procedure

Procedure: Adding connections for the IVR_Driver application

Prerequisites

Only the Message Server can be defined (if needed) as a connection for the IVR_Driver application.

Purpose: To add connections for the IVR_Driver application.

Start of procedure

1. In the IVRDriver_810 Properties dialog box, click the Connections tab.

2. Click Add to add a connection. The New Connection Info Properties dialog box appears.
3. Click the Folder icon next to the Server box to open a Browse dialog box that displays a list of server applications.
4. Select the Message Server and click OK. You are returned to the Connection Info Properties dialog box.
5. In the Connection Protocol, Local Timeout, Remote Timeout, and Trace Mode boxes, accept the default values.
6. Click OK to create the Message Server connection.

End of procedure

Setting up the I-Server Application

This section describes how to set up the I-Server application. It contains the following subsections:

- “Importing the I-Server application template” on [page 110](#)
- “Defining the I-Server application” on [page 111](#)
- “Configuring the I-Server application” on [page 111](#)

Procedure:

Importing the I-Server application template

Purpose: To import the I-Server application template into Configuration Manager.

Start of procedure

1. In Configuration Manager, expand Environment, right-click Application Templates, and select Import Application Template.
2. Navigate to the directory that contains the I-Server template file (I-Server_810.apd), select the file, and click Open. The New Application Template Properties dialog box appears.
3. In the Name box, accept the default value, or enter a new name for the I-Server template.
4. In the Type box, verify that IVR Server is selected.
5. In the Version box, verify that the correct version number is selected.
6. Click OK to import the I-Server application template.

End of procedure

Next Steps

You are now ready to define the I-Server application.

Procedure:
Defining the I-Server application

Purpose: To define the I-Server application in Configuration Manager.

Start of procedure

1. Expand Environment, right-click Applications, and select New > Application. A list of the available application templates is displayed.
2. Select the I-Server application template that you defined in [Procedure: Importing the I-Server application template](#), on [page 110](#), and then click OK. The New I-Server_810 Properties dialog box appears, with the General tab displayed.
3. In the Name box, accept the default value, or enter a new name for the I-Server application.
4. Click OK to create the I-Server application and close the New I-Server_810 Properties dialog box; or click Apply to create the application without closing the dialog box, so that you can perform additional actions.

End of procedure**Next Steps**

You are now ready to configure the I-Server application.

Configuring the I-Server application

The following sections describe how to configure the I-Server application in Configuration Manager:

- “Assigning an I-Server host” on [page 112](#)
- “Defining start parameters for the I-Server application” on [page 112](#)
- “Adding Connections for the I-Server Application” on [page 113](#)

Procedure: Assigning an I-Server host

Prerequisites

The I-Server and TServer_IVR applications must be configured on the same host.

Purpose: To assign an I-Server host.

Start of procedure

1. In the I-Server_810 Properties dialog box, click the Server Info tab.
2. In the Host box, select the host on which the I-Server application is installed.

Note: In order for a host to appear for selection in the drop-down list, you must predefine it by doing one of the following:

- Clicking the Folder icon next to the Host box.
 - In the main Configuration Manager window, select Environment > Host.
-

3. In the Ports box, add a new port number or edit an existing one.
4. In the Reconnect Timeout and Reconnect Attempts boxes, accept the default values.
5. Click Apply to assign the I-Server host.

End of procedure

Procedure: Defining start parameters for the I-Server application

Purpose: To define the start parameters for the I-Server application.

Start of procedure

1. In the I-Server_810 Properties dialog box, click the Start Info tab.
2. Enter a single period (.) in the following boxes:
 - Working Directory
 - Command Line

- ♦ Command Line Arguments

Note: These three boxes will be updated during the installation process.

3. In the Startup and Shutdown boxes, accept the default values.
4. Make sure that the Auto-Restart check box is not selected.
5. Click Apply to define the start parameters.

End of procedure

Procedure: Adding Connections for the I-Server Application

You must define the TServer_IVR application as a connection for the I-Server application. You can also define a connection to the following:

- T-Server (for IVR-Behind-Switch only)
- Message Server (for centralized logging)
- Stat Server

This section describes how to add all these connections.

Prerequisites

Before you can add a connection to the TServer_IVR application, you must import it (see “Importing the TServer_IVR application template” on [page 114](#)) and define it (see “Defining the TServer_IVR application” on [page 114](#)).

Start of procedure

1. In the I-Server_810 Properties dialog box, click the Connections tab.
2. Click Add to add a connection. The New Connection Info Properties dialog box appears.
3. Click the Folder icon next to the Server box to open a Browse dialog box that displays a list of server applications.
4. Select the TServer_IVR application, and then click OK. You are returned to the New Connection Info Properties dialog box.
5. In the Connection Protocol, Local Timeout, Remote Timeout, and Trace Mode boxes, accept the default values.
6. Click OK to create the connection.

7. To create connections to the T-Server, Message Server, and Stat Server applications (if necessary), repeat [Steps 2](#) through [Steps 6](#), selecting the appropriate application in [Step 4](#).

End of procedure

Setting up the TServer_IVR Application

This section describes how to set up the TServer_IVR application. It contains the following subsections:

- “Importing the TServer_IVR application template” on [page 114](#)
- “Defining the TServer_IVR application” on [page 114](#)
- “Configuring the TServer_IVR Application” on [page 115](#)

Procedure:

Importing the TServer_IVR application template

Purpose: To import the TServer_IVR application template into Configuration Manager.

Start of procedure

1. In Configuration Manager, expand Environment, right-click Application Templates, and select Import Application Template.
2. Navigate to the directory that contains the TServer_IVR template file (TServer_IVR_810.apd), select the file, and click Open. The New Application Template Properties dialog box appears.
3. In the Name box, accept the default value, or enter a new name for the TServer_IVR template.
4. In the Type box, verify that T-Server is selected.
5. In the Version box, verify that the correct version number is selected.
6. Click OK to import the TServer_IVR application template.
7. You are now ready to define the TServer_IVR application.

End of procedure

Procedure:

Defining the TServer_IVR application

Purpose: To define the TServer_IVR application in Configuration Manager.

Start of procedure

1. Expand Environment, right-click Applications, and select New > Application. A list of the available application templates is displayed.
2. Select the TServer_IVR application template that you defined in “Importing the TServer_IVR application template” on [page 114](#), and then click OK. The New TServer_IVR_810 Properties dialog box appears, with the General tab displayed.
3. In the Name box, accept the default value, or enter a new name for the TServer_IVR application.

Note: If the name includes blank spaces, you must enclose it in double quotation marks. For more information, see [Step 19](#) on [page 152](#).

4. Click OK to create the TServer_IVR application and close the New TServer_IVR_810 Properties dialog box; or click Apply to create the application without closing the dialog box, so that you can perform additional actions.

End of procedure**Next Steps**

You are now ready to configure the TServer_IVR application.

Configuring the TServer_IVR Application

The following sections describe how to configure the TServer_IVR application in Configuration Manager:

- “Creating a virtual switching office” on [page 115](#)
- “Creating a virtual switch” on [page 116](#)
- “Assigning a virtual switch” on [page 116](#)
- “Assigning a TServer_IVR host” on [page 117](#)
- “Defining start parameters for the TServer_IVR application” on [page 118](#)
- “Enabling network logging” on [page 118](#)

Procedure:**Creating a virtual switching office**

Purpose: You must create a virtual switching office in order to configure the TServer_IVR application.

Start of procedure

1. Expand Environment, right-click Switching Offices, and select New > Switching Office. The New Switching Office Properties dialog box appears.
2. In the Name box, enter Virtual Switching Office.
3. In the Switch Type box, select Virtual Switch for IVR In-Front.
4. Click OK to create the virtual switching office.

End of procedure

**Procedure:
Creating a virtual switch**

Purpose: To create a virtual switch for the TServer_IVR application:

Start of procedure

1. Expand Resources, right-click Switches, and select New > Switch. The New Switch Properties dialog box appears.
2. In the Name box, enter Virtual_Switch_810.
3. In the Switching Office box, select Virtual Switching Office.
4. In the T-Server box, select None.
5. In the DN Range box, do one of the following, depending on the IVR Server configuration mode:
 - IVR In-Front: Enter one DN for every IVR port.
 - IVR-Behind-Switch: Specify nothing.
6. Click OK to create the virtual switch.

End of procedure

**Procedure:
Assigning a virtual switch**

Purpose: To assign a virtual switch to the TServer_IVR application.

Start of procedure

1. In the TServer_IVR_810 Properties dialog box, click the Switches tab.
2. Click Add to open a Browse dialog box that displays a list of switches.

3. Select a virtual switch, and then click OK. You are returned to the TServer_IVR_810 Properties dialog box, which now displays the virtual switch that you added.
4. Click Apply to assign the virtual switch to the TServer_IVR application.

End of procedure

Procedure: Assigning a TServer_IVR host

Prerequisites

The TServer_IVR and I-Server applications must be configured on the same host.

Purpose: To assign a TServer_IVR host.

Start of procedure

1. In the TServer_IVR_810 Properties dialog box, click the Server Info tab.
2. In the Host box, select the host on which the TServer_IVR application is installed.

Note: In order for a host to appear for selection in the drop-down list, you must predefine it by doing one of the following:

- Clicking the Folder icon next to the Host box.
 - In the main Configuration Manager window, select Environment > Host.
-

3. In the Ports box, add a new port number or edit an existing one.
4. In the Backup Server box, enter the name of your backup server if you want to use Warm or Hot Standby mode; otherwise, accept the default value ([None]).
5. In the Redundancy Type box, select Warm Standby or Hot Standby (as applicable), or accept the default value (Not Specified).
6. In the Reconnect Timeout and Reconnect Attempts boxes, accept the default values.
7. Click Apply to assign the TServer_IVR host.

End of procedure

Procedure: Defining start parameters for the TServer_IVR application

Purpose: To define start parameters for the TServer_IVR application.

Start of procedure

1. In the TServer_IVR_810 Properties dialog box, click the Start Info tab.
2. Enter a single period (.) in the following boxes:
 - Working Directory
 - Command Line
 - Command Line Arguments

Note: These three boxes will be updated during the installation process.

3. In both the Startup and Shutdown boxes, Genesys recommends that you set the value to 600.
4. Select the Auto-Restart check box. This setting restarts the TServer_IVR application if it closes abnormally, or if it is stopped without using the Solution Control Interface (SCI).

If Auto-Restart is selected, and you stop the TServer_IVR application without using the SCI, Genesys Framework (and the Local Control Agent [LCA]) will restart it. Therefore, if you want to stop the TServer_IVR application without using the SCI, clear the Auto-Restart check box.

Note: Make sure that you set the Startup box to a value that allows enough time for the TServer_IVR application to fully start. If the time is exceeded, the TServer_IVR application is terminated, and it is not restarted, even if Auto-Restart has been selected.

5. Click OK to define the start parameters.

End of procedure

Procedure: Enabling network logging

Prerequisites

Before you can configure IVR Server for centralized logging, you must define the following in Configuration Manager:

- A Message Server and corresponding Database Access Point (DAP).
- A Log DB Server.

For information about defining a Message Server and a DAP, see *Framework 8.1 Configuration Manager Help* and the *Framework 8.1 Deployment Guide*.

Purpose: To enable Network Logging, you must add a connection between the TServer_IVR application and the Message Server.

Start of procedure

1. In Configuration Manager, right-click the TServer_IVR application, and then select Wizard > Configure. The TServer_IVR Properties dialog box appears.
2. On the General tab, click Run Log Wizard.
3. Follow the on-screen prompts, making sure that you select Network Log Server on the Log Outputs screen.

The Log Wizard creates the Message Server connection for you.

End of procedure

Setting up the TServer_IVR_Network Application

This section describes how to set up the TServer_IVR_Network application. It contains the following subsections:

- “Importing the TServer_IVR_Network application template” on [page 119](#)
- “Defining the TServer_IVR_Network application” on [page 120](#)
- “Configuring the TServer_IVR_Network Application” on [page 121](#)

Procedure:

Importing the TServer_IVR_Network application template

Purpose: To import the TServer_IVR_Network application template into Configuration Manager.

Start of procedure

1. In Configuration Manager, expand Environment, right-click Application Templates, and select Import Application Template.
2. Navigate to the directory that contains the TServer_IVR_Network template file (TServer_IVR_Network_810.apd), select the file, and click Open. The New Application Template Properties dialog box appears.
3. In the Name box, accept the default value, or enter a new name for the TServer_IVR_Network template.
4. In the Type box, verify that T-Server is selected.
5. In the Version box, verify that the correct version number is selected.
6. Click OK to import the TServer_IVR_Network application template.

End of procedure**Next Steps**

You are now ready to define the TServer_IVR_Network application.

Procedure:
Defining the TServer_IVR_Network application

Purpose: To define the TServer_IVR_Network application in Configuration Manager.

Start of procedure

1. Expand Environment, right-click Applications, and select New > Application. A list of the available application templates is displayed.
2. Select the TServer_IVR_Network application template that you defined in “Importing the TServer_IVR_Network application template” on [page 119](#), and then click OK. The New TServer_IVR_Network Properties dialog box appears, with the General tab displayed.
3. In the Name box, accept the default value, or enter a new name for the TServer_IVR_Network application.

Note: If the name includes blank spaces, you must enclose it in double quotation marks. For more information, see [Step 19](#) on [page 158](#).

4. Click OK to create the TServer_IVR_Network application and close the New TServer_IVR_Network Properties dialog box; or click Apply to create the application without closing the dialog box, so that you can perform additional actions.

End of procedure

Next Steps

You are now ready to configure the TServer_IVR_Network application.

Configuring the TServer_IVR_Network Application

The following sections describe how to configure the TServer_IVR_Network application in Configuration Manager:

- “Creating a network switching office” on [page 121](#)
- “Creating a network switch” on [page 122](#)
- “Assigning a network switch” on [page 122](#)
- “Assigning an IVR host” on [page 122](#)
- “Defining start parameters for the TServer_IVR_Network application” on [page 123](#)
- “Enabling network logging” on [page 124](#)

Procedure:

Creating a network switching office

Purpose: You must create a network switching office in order to configure the TServer_IVR_Network application.

Start of procedure

1. Expand Environment, right-click Switching Offices, and select New > Switching Office. The New Switching Office Properties dialog box appears.
2. In the Name box, enter Network Switching Office.
3. In the Switch Type box, select GenSpec XML.
4. Click OK to create the network switching office.

End of procedure

Procedure: Creating a network switch

Purpose: To create a network switch for the TServer_IVR_Network application.

Start of procedure

1. Expand Resources, right-click Switches, and select New > Switch. The New Switch Properties dialog box appears.
2. In the Name box, enter Network Switch.
3. In the Switching Office box, select Network Switching Office.
4. In the T-Server box, select None.
5. Click OK to create the network switch.

End of procedure

Procedure: Assigning a network switch

Purpose: To assign a network switch to the TServer_IVR_Network application.

Start of procedure

1. In the TServer_IVR_Network Properties dialog box, click the Switches tab
2. Click Add to open a Browse dialog box that displays a list of switches.
3. Select IVR Network Switch, and then click OK. You are returned to the TServer_IVR_Network Properties dialog box, which now displays the network switch that you added.
4. Click Apply to assign the switch to the TServer_IVR_Network application.

End of procedure

Procedure: Assigning an IVR host

Purpose: To assign an IVR host to the TServer_IVR_Network application.

Start of procedure

1. In the TServer_IVR_Network Properties dialog box, click the Server Info tab.
2. In the Host box, select the host on which the TServer_IVR_Network application is installed.

Note: In order for a host to appear for selection in the drop-down list, you must predefine it by doing one of the following:

- Clicking the Folder icon next to the Host box.
 - In the main Configuration Manager window, select Environment > Host.
-

3. In the Ports box, add a new port number or edit an existing one.
4. In the Backup Server box, enter the name of your backup server if you want to use Warm Standby or Hot Standby modes; otherwise, accept the default value ([None]).
5. In the Redundancy Type box, select Warm Standby or Hot Standby (as applicable), or accept the default value (Not Specified).
6. In the Reconnect Timeout and Reconnect Attempts boxes, accept the default values.
7. Click Apply to assign the TServer_IVR_Network host.

End of procedure

Procedure: **Defining start parameters for the TServer_IVR_Network application**

Purpose: To define start parameters for the TServer_IVR_Network application.

Start of procedure

1. In the TServer_IVR_Network Properties dialog box, click the Start Info tab.
2. Enter a single period (.) in the following boxes:
 - Working Directory
 - Command Line
 - Command Line Arguments

Note: These three boxes will be updated during the installation process.

3. In both the Startup and Shutdown boxes, Genesys recommends that you set the value to 600.
4. Select the Auto-Restart check box. This setting restarts the TServer_IVR_Network application if it closes abnormally, or if it is stopped without using the SCI.

If Auto-Restart is selected, and you stop the TServer_IVR application without using the SCI, Genesys Framework (and the LCA) will restart it. Therefore, if you want to stop the TServer_IVR_Network application without using the SCI, clear the Auto-Restart check box.

Note: Make sure that you set the Startup box to a value that allows enough time for the TServer_IVR_Network application to fully start. If the time is exceeded, the TServer_IVR_Network application is terminated, and it is not restarted, even if Auto-Restart has been selected.

5. Click OK to define the start parameters.

End of procedure

Procedure: Enabling network logging

Prerequisites

Before you can configure IVR Server for centralized logging, you must define the following in Configuration Manager:

- A Message Server and corresponding DAP.
- A Log DB Server.

For information about defining a Message Server and a DAP, see *Framework 8 Configuration Manager Help* and the *Framework 8 Deployment Guide*.

Purpose: To enable Network Logging, by adding a connection between the TServer_IVR_Network application and the Message Server.

Start of procedure

1. In Configuration Manager, right-click the TServer_IVR_Network application, and then select Wizard > Configure. The TServer_IVR_Network Properties dialog box appears.
2. On the General tab, click Run Log Wizard.

3. Follow the on-screen prompts, making sure that you select Network Log Server on the Log Outputs screen.

The Log Wizard creates the Message Server connection for you.

End of procedure

5

Installing IVR Server

This chapter describes the standard installation procedure for an IVR Server application on the UNIX and Windows operating systems. It contains the following sections:

- [Installing on UNIX, page 127](#)
- [Installing on Windows, page 128](#)

Note: You can install IVR Server on any computer that belongs to the site where the IVR Interface Option 8.1 product is used (including the computer on which the IVR Driver is installed). However, the operating system of the host on which you install the IVR must match the operating system on which the IVR Server was built.

For more information about installing server applications on the UNIX and Windows operating systems, see the *Framework 8.1 Deployment Guide*.

Installing on UNIX

The installation process for UNIX-oriented operating systems uses a new method for identifying the license file that is to be used for IVR Server. For migration purposes, IVR Server still supports the previous method of specifying the directory path (through the `-l` command-line parameter). Furthermore, the Installation Wizard still includes the following prompt: Please enter the full path to the license file. However, Genesys recommends that you use the new format `portnumber@hostname`, to specify the license file. For a description, see the *Genesys Licensing Guide*.

For UNIX-oriented operating systems, when you enter a value in the format `portnumber@hostname`, the Installation Wizard displays the following message: This is an invalid path. Do you want to enter another path? Answer No to this prompt, and the installation continues and completes successfully.

For UNIX-oriented operating systems, the Installation Wizard creates a *.sh file (for example, run.sh), which presents the prompted information in the following format:

```
-host <cfgSrvr_Host> -port <cfgSrvr_CommPort> -app  
"<AppName_defined_nCME>" -l "<license_path>"
```

If this information is accurate, the *.sh file will execute correctly.

Procedure: Installing a Genesys IVR application on UNIX

Purpose: To install a Genesys IVR Server application on UNIX.

Start of procedure

1. Load the DVD and locate the shell script called INSTALL.SH.
2. Run the script from the command line by typing sh and the file name, as follows:
sh INSTALL.SH
3. Specify the host name of the computer on which IVR Server is to run.
4. When prompted, specify the:
 - Host name of the computer on which Configuration Server is running.
 - Port that client applications use to connect to Configuration Server.
 - User name and password that are used to log in to the Configuration Layer.
5. From the list of applications, type the number corresponding to the IVR Server application that you want to install.
6. Specify the destination directory and full path for the IVR Server application.
7. Choose either the 32-bit or the 64-bit option, if prompted, according to your environment.

End of procedure

Installing on Windows

Use the procedure below to install Genesys IVR on Windows.

Procedure: **Installing a Genesys IVR application on Windows**

Purpose: To install a Genesys IVR Server application on Windows.

Start of procedure

1. Load the DVD and run `Setup.exe`.
2. Specify the host and port of Configuration Server. Accept `ITCUtility` as the name of the Installation Configuration Utility application.
3. Log in to the Configuration Layer and confirm the host name of the computer on which IVR Server is to run.
4. From the list of IVR applications displayed, select the IVR Server application to install.
5. Specify the destination directory for the IVR Server application.
6. Specify the program folder for the IVR Server application.
7. Decide whether you want to install this server as a Windows Service. For more information, see the *Framework 8.1 Deployment Guide*.
8. When icons for this server appear, click `Finish`.

End of procedure

6

Wizard Configuration

This chapter describes how to use the Genesys IVR Interface Option Wizard to configure IVR Server, IVR Drivers, and other required objects.

Note: If you want to use the IVR Network T-Server configuration mode, you can use either the wizard, or the manual configuration procedures described in Chapter 7 on [page 135](#).

This chapter contains the following sections:

- [Before Using the Wizard, page 131](#)
- [Using the Wizard in Configuration Manager, page 132](#)
- [Using the Wizard Manager, page 133](#)

Note: For example configurations, see Appendix A on [page 263](#).

Before Using the Wizard

Before you can use the Genesys 8 IVR Interface Option Wizard, you must complete the following tasks:

1. Install the Genesys Wizard Manager from the Management Framework product DVD, if it has not already been installed in your environment. It must be installed on the same machine as Configuration Manager, but not necessarily on the same machine as the IVR Server.
2. Install the IVR Interface Option Wizard from the IVR Server product DVD. It must be installed on the same machine as the Genesys Wizard Manager and Configuration Manager.
3. Collect the information that you will be asked to enter when you run the wizard. This information is described in “Before You Configure” on [page 136](#).

4. Enable the correct licenses for IVR Server. For more information, see “license Section” on [page 187](#), and the *Genesys Licensing Guide*.

Using the Wizard in Configuration Manager

If you are migrating from a previous release of IVR Interface Option, you can use the following wizard instructions to associate the objects that you have already defined in Configuration Manager with the 8.1 releases of the IVR Server and IVR Driver.

For example, although you must create a new object for the `IVR_Driver` application, you can upgrade your existing `I-Server`, `TServer_IVR`, `IVR`, and other objects for use with IVR Interface Option 8.1.

After you have created new objects for IVR Interface Option 8.1, or upgraded existing pre-8.1 objects, you can use the IVR Interface Option Wizard in `configure` mode to modify those objects. For example, you can:

- Add or modify secondary I-Server applications for Load Balancing.
- Configure ranges of IVR ports with DNS and agent logins (see “[Creating a range of IVR Ports](#)”).
- Define statistics.
- Add or modify connections to premise T-Servers.

If you want to preserve your pre-8.1 objects, complete the steps in “Using the Wizard Manager” on [page 133](#) to create new objects for release 8.1.

Note: Because the IVR Interface Option Wizard does not support the `TServer_IVR_Network` application, you can only create a `TServer_IVR_Network` application manually in Configuration Manager.

Procedure:

Upgrading IVR Interface Option objects to release 8.1

Purpose: To upgrade existing IVR Interface Option pre-8.1 objects to release 8.1 in Configuration Manager.

Start of procedure

1. Right-click the 6.5, 7.0, 7.1, 7.2, 7.5, or 8.0 IVR object name, and then select `Wizard > Configure`. The object’s `Properties` dialog box appears.
2. Click `upgrade` to start the IVR Interface Option Wizard in `configure` mode.
3. Follow the on-screen prompts to upgrade the object.

End of procedure

Procedure: Creating a range of IVR Ports

Purpose: To create a range of IVR ports with associated DNs and agent logins in Configuration Manager.

Start of procedure

1. Right-click the appropriate IVR object, and then select **Wizard > Configure**. The **IVR Properties** dialog box appears.
2. Click the **IVR Ports** tab.
3. Click **New Range** to start the IVR Ports Range Wizard.
4. Follow the on-screen prompts to create a range of IVR ports.

End of procedure

Using the Wizard Manager

Use the Genesys Wizard Manager to create new objects in Configuration Manager for the 8 release of the IVR Server and IVR Driver. These new objects, which can be used with IVR Interface Option 8.1, include:

- I-Server application
- TServer_IVR application
- IVR_Driver application
- TServer_IVR_Network application.

Note: This application object is required *only* for the IVR Network T-Server configuration mode.

- IVR object (including a range of ports with assigned DNs and agent logins)
- Switch object
- Switching Office object

Procedure: Configuring new IVR Interface Option 8.1 objects

Purpose: To configure new IVR Interface Option 8.1 objects in Configuration Manager.

Start of procedure

1. From the Windows Start menu, select Genesys Solutions > Genesys Wizards Manager > Genesys Wizard Manager to start the Genesys Wizard Manager.
2. Log in to open the Genesys Wizard Manager main window.
3. On the left pane, select IVR. The right pane displays the IVR screen.
4. Select Deploy Genesys IVR Interface Option in your contact center to start the IVR Interface Option Wizard.
5. Click Next.
6. Follow the on-screen prompts to create new configuration objects for use with IVR Interface Option 8.1.

End of procedure

Wizard Manager Usage Notes

- Note 1** On some wizard screens, to enter an application name in a box, you must click the Folder icon next to that box. In some cases, you can use the Browse for Application dialog box that appears in two ways:
- Select from among the existing applications that are displayed in the dialog box.
 - Create a new application by clicking the New Application icon at the top of the dialog box.
- Note 2** When you select an existing pre-8.1 I-Server application in the Genesys Wizard Manager, the wizard upgrades it to release 8.1. If you upgrade an existing pre-8.1 I-Server application, you must upgrade the corresponding TServer_IVR application. If you choose to create a new release 8.1 I-Server application instead, you must also create a new TServer_IVR application.
- Note 3** During configuration of the I-Server application, the wizard opens a dialog box in which you can specify statistics for the I-Server to monitor. Before I-Server can monitor these statistics, they must be defined in the TimeProfiles section of the Options tab for the Stat Server application. For more information on configuring TimeProfiles options, see the *Framework 8.1 Stat Server User's Guide*.

7

Manual Configuration

This chapter describes how to manually configure the IVR Server, IVR Drivers, and other required objects. It also lists the information that you need to gather before you perform the configuration tasks.

You can also use the IVR Interface Option 7.5 Wizard (see Chapter 6 on [page 131](#)) to create and configure most of the required objects in Configuration Manager. Even if you use the wizard, you can use the procedures in this chapter to add individual objects, or to change existing objects.

Note: If you want to use the IVR Network T-Server configuration mode, you must use the manual configuration procedures in this chapter, because the IVR Interface Option Wizard does not support this configuration mode.

This chapter contains the following sections:

- [Before You Configure, page 136](#)
- [Required Configuration Tasks, page 139](#)
- [Logging In, page 141](#)
- [Enabling the Annex Tab, page 141](#)
- [Configuring the Switching Office, page 142](#)
- [Configuring Switches, page 143](#)
- [Configuring DNSs, page 145](#)
- [Configuring IVRs, page 146](#)
- [Configuring IVR Ports, page 147](#)
- [Configuring the I-Server Application, page 150](#)
- [Configuring the TServer_IVR Application, page 152](#)
- [Configuring the IVR_Driver Application, page 156](#)
- [Configuring the TServer_IVR_Network Application, page 158](#)
- [Configuring Agent Logout for Load-Sharing IVR Servers, page 162](#)
- [Adding Servers, page 163](#)

For example configurations, refer to Appendix A on [page 263](#).

Before You Configure

This section describes the information that you will need in order to successfully complete the IVR Server configuration tasks that are described in the rest of this chapter. Genesys recommends that you gather this information before you begin configuration.

Also, before you can configure IVR Server, you must enable the correct licenses for it. For more information, see “license Section” on [page 187](#) and the *Genesys Licensing Guide*.

Note: Until all components are installed, each configured application appears in Configuration Manager as Disabled, without a template assigned to it, and with Unknown Application Type displayed on the General tab of its Properties dialog box. After you install them, each application’s State changes to Enabled, a correct template is assigned to it, and the correct application type is displayed on the General tab.

Preconfiguration Notes

This section contains special instructions that apply to IVR Server general releases in the following cases:

1. You are adding IVR ports in Configuration Manager, and you need to know what DN type to use, regardless of the T-Server release.
2. You are upgrading the T-Server from release 6.x to 7.x, and you need to order a new license file.

This section discusses the following topics:

- Which DN type to use when configuring IVR ports in Configuration Manager.
- How many agent seat and technical DN licenses are required for each IVR port if a T-Server is used.

(Starting with 7.x release, T-Servers implemented a new licensing methodology in which you must determine the number of agent seat and technical DN licenses for each IVR port.)

Note: The configuration of the IVR ports directly affects these numbers.

IVR Port Configuration Exceptions

Although Genesys generally recommends that you configure IVR ports in Configuration Manager as Voice Treatment Ports, there are exceptions, and in these cases you must configure them as ACD Positions, or as ACD Position/Extension pairs. In order to configure the IVR ports correctly, it is important that you understand these exceptions.

IVR Ports and Switch Configuration

There are two different ways to configure a computer-telephony integration (CTI)-controlled IVR port on the switch:

- As a regular DN that is always ready to receive calls.
- As a DN that requires an agent login in order to receive calls.

In some cases, you can also associate an IVR port with both a regular DN and an agent login DN.

IVR Ports and Genesys Configuration Manager Configuration

If Genesys Universal Routing Server (URS) will be distributing calls to IVR ports for treatment, you should enter all IVR ports in Configuration Manager and add them to Places. URS will distribute calls to IVR ports for treatment based on the availability of the Places. Stat Server determines Place availability, using an algorithm that depends greatly on the Configuration Manager DN type. For more information, see [“Stat Server Place Status Algorithm.”](#)

[Table 10](#) contains recommendations for what DN type to use when configuring an IVR port in Configuration Manager, depending on the way in which the IVR port is configured on the switch. It also outlines the T-Server 7.x license requirements for each configuration type.

Table 10: DN Type Recommendations

IVR port configuration on the switch	Recommended DN Type in CME	Required T-Server 7.x license type and amount
A regular DN that is always ready to receive calls.	Voice Treatment Port	One technical DN license
A DN that requires an agent login in order to receive calls.	ACD Position	One agent seat license
A DN that requires an agent login and a regular DN (an agent login happens at the former and calls are distributed to the latter).	ACD Position and Extension	Two agent seat licenses

Stat Server Place Status Algorithm

This section explains how Stat Server determines a Place Status, in order to clarify the reasoning behind the IVR port configuration rules. In the following discussion, an *IVR Port Place* is defined as a Place in CME that includes a DN that represents an IVR port. In some cases—for example, for switches such as Nortel Meridian—an IVR Port Place can include two DNs, representing an analog ACD Position and an Extension.

Out of the rules that Stat Server uses to determine a Place Status, the following apply to IVR Port Places:

1. Generally, a Place inherits its Status from the DN that is linked to it. In other words, the Place Status is the same as the Status of the DN that is linked to the Place. If more than one DN is linked to the Place, and if these DNs have different Statuses, the Place inherits the DN Status with the highest priority, according to the Agent Status Priority Table.

For example, suppose that an ACD Position and an Extension are linked to the Place. If the ACD Position Status is `WaitForNextCall`, and the Extension Status is `CallInbound`, the overall Place Status will be `CallInbound`.

2. If there is an Agent Login at the DN that is associated with the Place (at least one of the DNs), Rule 1 applies.
3. If there is no Agent Login at any of the DNs that are associated with the Place, Rule 1 applies, with the following exception: If the resulting Status, according to Rule 1, is `WaitForNextCall`, Stat Server sets the Place Status to `NotReadyForNextCall`.

For example, suppose that an Extension is linked to the Place, and that there is no Agent Login. If the Extension Status is `CallInbound`, the overall Place Status will be `CallInbound`. However, if the Extension Status is `WaitForNextCall`, the overall Place Status will be `NotReadyForNextCall`.

4. If there is no Agent Login at the DN that is associated with the Place, but the DN type is `Voice Treatment Port`, Rule 1 applies.

For example, suppose that a Voice Treatment Port is linked to the Place, and that there is no Agent Login. If the Voice Treatment Port Status is `WaitForNextCall`, the overall Place Status will be `WaitForNextCall`.

For the complete set of IVR port configuration rules, see the *Framework 8.1 Stat Server User's Guide*.

Configuration Tasks

1. Switching Office name
2. Switch name
3. DNs on the switch:
 - Extensions

- Routing Points (if required)
 - ACD Queues (if required)
4. IVR name
 5. Key/Value pairs to be returned to clients at login (if required)
 6. Login response data
 7. IVR Ports, and how they map to the DNs on the switch
 8. Auto login information for each IVR port (if required):
 - Agent ID
 - Queue
 - Password
 - SetReady
 - SetLoggedIn
 9. IVR Server application name
 10. Statistics information for each Stat (if required):
 - Statistic name (as specified in network messages)
 - obj_id
 - obj_type
 - server_name
 - stat_type
 - time_profile
 - update_frequency
 11. TServer_IVR application name
 12. Host/port for the TServer_IVR application
 13. GLI Server host/port
 14. UseQueue option, if you will be using external routing

Required Configuration Tasks

Table 11 shows the required configuration tasks for each IVR Server configuration mode.

Table 11: Task Summary: Configuring IVR Server

Required Configuration Tasks	IVR-In-Front	IVR-Behind-Switch	IVR Network T-Server	Configuration Notes
Log in (page 141)	X	X	X	
Enable the Annex tab (page 141)	X	X	X	Enables the Annex tab for all objects that offer it.

Table 11: Task Summary: Configuring IVR Server (Continued)

Required Configuration Tasks	IVR-In-Front	IVR-Behind-Switch	IVR Network T-Server	Configuration Notes
Configure the switching office (page 142)	X	X	X	<ul style="list-style-type: none"> IVR-In-Front mode—Associate the switching office with a virtual switch. IVR-Behind-Switch and IVR Network T-Server modes—Associate the switching office with a physical switch.
Configure switches (page 143)	X	X	X	<ul style="list-style-type: none"> IVR-In-Front mode—Create a virtual switch and associate it with a TServer_IVR application (instead of a premise T-Server). IVR-Behind-Switch mode—Create a physical switch and associate it with a premise T-Server. IVR Network T-Server mode—Create a physical switch and associate it with a TServer_IVR_Network application.
Configure DNs (page 145)	X	X	X	
Configure IVRs (page 146)	X	X	X	
Configure IVR ports (page 147)	X	X	X	
Configure the I-Server application (page 150)	X	X		The I-Server application is not used in IVR Network T-Server mode, and therefore should not be configured for it.
Configure the TServer_IVR application (page 152)	X	X		The TServer_IVR application is not used in IVR Network T-Server mode, and therefore should not be configured for it. Instead, use the TServer_IVR_Network application.

Table 11: Task Summary: Configuring IVR Server (Continued)

Required Configuration Tasks	IVR-In-Front	IVR-Behind-Switch	IVR Network T-Server	Configuration Notes
Configure the IVR_Driver application (page 156)	X	X	X	The IVR_Driver application is required in all modes, unless you are using IVR Driver 7 for Aspect, CONVERSANT, or Microsoft Speech Server. The IVR_Driver application is not available for use with these IVR Drivers.
Configure the TServer_IVR_Network application (page 158)			X	The TServer_IVR_Network application is required <i>only</i> for IVR Network T-Server mode. In other modes, use the TServer_IVR application instead.

Logging In

Note: This guide describes how to define and configure the required IVR Interface Option 8.1 objects by using the Configuration Manager component of Genesys Framework 8.1. Other releases of Genesys Framework might also be supported. For information about the supported releases, see the *Genesys Migration Guide*. For a detailed description of Configuration Manager, and how to create a new application, see the *Framework 8.1 Configuration Manager Help* and the *Framework 8.1 Deployment Guide*.

Start Configuration Manager and log in.

Enabling the Annex Tab

Note: You must enable the Annex tab in order to successfully configure IVR Server in Configuration Manager.

Procedure: **Enabling the Annex tab**

Purpose: To enable the Annex tab in the Properties dialog box for all applicable objects in Configuration Manager.

Start of procedure

1. In the Configuration Manager main window, select **View > Options**. The Options dialog box appears.
2. Select the **Show Annex tab in object properties** check box.
3. Click **OK**.

End of procedure

Configuring the Switching Office

You must configure the switching office at this point only if your environment uses premise T-Servers, and if they have not already been configured.

Procedure: **Configuring the Switching Office**

Purpose: To configure the switching office in Configuration Manager.

Start of procedure

1. Expand **Environment**, right-click **Switching Offices**, and select **New > Switching Office**. The **New Switching Office Properties** dialog box appears.
2. In the **Name** box, enter a name for the switching office. This name is required, and it must be unique within the Configuration Database.
3. In the **Switch Type** box, select a switch type. After it is specified, the type cannot be changed.
4. Select the **State Enabled** check box to indicate that the object is in regular operating condition and can be used without restriction.
5. Click **OK** to complete the configuration of the switching office.

End of procedure

Configuring Switches

A *switch* is an aggregate of telephony resources within a switching office, controlled through one T-Server or TServer_IVR application. When a new switch is registered in the Configuration Database, the Configuration Layer automatically creates two folders under it, one for DNs, and one for Agent Logins.

Procedure: Configuring Switches

Purpose: To configure a switch in Configuration Manager.

Start of procedure

1. Expand Resources (or Tenant Name), right-click Switches, and select New > Switch. The New Switch Properties dialog box appears.
2. In the Name box, enter a name for the switch. This name is required, and it must be unique within the switching office or tenant.
3. In the Switching Office box, select the switching office. Once specified, the switching office cannot be changed.
4. In the T-Server box, select the TServer_IVR application through which the telephony objects of this switch are controlled. Each TServer_IVR can be associated with only one switch.
5. In the Switch Type box, select a switch type. Once specified, the type cannot be changed.
6. Select the State Enabled check box to indicate that the object is in regular operating condition and can be used without restriction.
7. If you want to add or edit an access code, see [“Using Access Codes.”](#)
8. Click OK to complete the configuration of the switch.

End of procedure

Using Access Codes

Procedure: Adding/editing access codes

Purpose: To add or edit an access code in Configuration Manager.

Start of procedure

1. Expand Resources (or Tenant Name), right-click Switches, and select New > Switch. The New Switch Properties dialog box appears.
2. Click either the Access Codes or the Default Access Codes tab.
3. Do one of the following:
 - To add an access code, click Add.
 - To edit an existing access code, select the code, and then click Edit.
4. The New Switch Access Code Properties dialog box appears.
5. In the Switch box, select the switch to which the access code is assigned. If this is a default access code, this box is not available.
6. In the Code box, enter the prefix that is used to reach DN's of the switch (specified in the Switch box) when placing or transferring calls from DN's of the switch that is to be configured.
7. In the Target Type box, select the type of target within the switch.
8. In the Route Type box, select the type of routing for the target (specified in the Target Type box) for this switch.
9. In the DN Source box, enter the source of the information that is used to specify the origination point in routing instructions.
10. In the Destination Source box, enter the source of the information that is used to specify the destination in routing instructions.
11. In the Location Source box, enter the source of the information that is used to specify the location in routing instructions.
12. In the DNIS Source box, enter the source of the information that is used to specify the DNIS in routing instructions.
13. In the Reason Source box, enter the source of the information that is used to specify the reasons in routing instructions.
14. In the Extension Source box, enter the source of the information that is used to specify the extensions in routing instructions.
15. Click OK to add or edit the access code.

End of procedure

Configuring DNs

Note: DNs that are configured must be of type Routing Point or Routing Queue.

Procedure: Configuring DNs

Purpose: To configure a DN in Configuration Manager.

Start of procedure

1. Expand Resources (or Tenant Name) > Switches > Switch, right-click DNs, and select New > DN. The New DN Properties dialog box appears.
2. In the Number box, enter the directory number that is assigned to this DN within the switch. This value is required, and it must be unique within the Configuration Database.
3. In the Type box, select the DN type. To use the IVR Server 7.x pool of licenses for Voice Treatment Ports, you must define DNs here with the Voice Treatment type.
4. In the Association box, enter an entity that is to be permanently associated with this DN—for example, an IVR port number, channel name, or access number.
5. In the Register box, select whether the Premise T-Server is to register this DN within the switch.

Note: This option must be set to `false` if you are configuring a Virtual Routing Point in IVR-Behind-Switch mode.

6. Select the State Enabled check box to indicate that the object is in regular operating condition and can be used without restriction.
7. Click OK to complete the configuration of the DN.

End of procedure

Configuring IVRs

IVRs (Interactive Voice Response devices) are telephony objects that consist of IVR ports, which are controlled through IVR Drivers. When a new IVR is registered in the Configuration Database, the Configuration Layer automatically creates an IVR Ports folder under it.

Note: If you have more than one IVR, you should plan a naming scheme for each of the corresponding IVR Drivers before you complete these configuration tasks.

Procedure: Configuring IVRs

Purpose: To configure IVRs in Configuration Manager.

Start of procedure

1. Expand Resources (or Environment), right click IVRs, and select New > IVR. The New IVR Properties dialog box appears.
2. In the Name box, enter the name of the IVR Driver. This value is required, and it must be unique within the Configuration Database (in an enterprise environment) or tenant (in a multi-tenant environment).
3. In the Description box, enter a brief description of the IVR.
4. In the Type box, select the type of this IVR. This value is required, and it cannot be changed if at least one IVR port is associated with this IVR. If your specific IVR type is not listed, select Unknown.
5. In the Version box, enter the version of the IVR. This value is required.
6. In the IVR Interface Server box, select the name of the application of the IVR Server type that serves this IVR. You can assign the IVR Interface Server to this driver by clicking the Folder icon.
7. Select the State Enabled check box to indicate that the object is in regular operating condition and can be used without restriction.
8. Click OK to complete the configuration of the IVR.

End of procedure

Configuring IVR Ports

IVR ports are telephony objects at which telephone calls can reside and be handled; they are uniquely identified by the numbers within IVRs.

Note: Each DN must be assigned to, at most, one IVR port.

The following subsections describe how to configure IVR ports in Configuration Manager:

- [Creating a Single IVR Port, page 147](#)
- [Configuring Multiple IVR Ports, page 148](#)
- [Configuring the Auto-Login Feature, page 150](#)

Creating a Single IVR Port

Procedure: Creating a single IVR port

Purpose: To create a single IVR port.

Start of procedure

1. Expand Resources (or Environment) > IVRs > [specific IVR], right click IVR Ports, and select New > IVR Port. The New IVR Port Properties dialog box appears.
2. In the Port Number box, enter the number of the IVR port. See [Table 12](#) for the starting port number to use for your IVR Driver.
3. In the Description box, enter a brief description of this IVR port.
4. In the Associated DN box, enter the DN associated with this IVR port.

Note: Do not configure two IVR ports to point to the same DN/switch combination.

5. Click the Folder icon next to the Associated DN box to define the association between the IVR port and a DN for the switch that is connected to this IVR. The IVR port/DN association is completed through additional steps.
6. Select the State Enabled check box to indicate that the object is in regular operating condition and can be used without restriction.

7. Click OK to create the IVR port.

End of procedure

Note: If your IVR ports must be able to log in as agents, see “Configuring the Auto-Login Feature” on [page 150](#).

Configuring Multiple IVR Ports

Procedure: Configuring multiple IVR ports

Purpose: To configure your IVR to manage multiple telephony ports.

Start of procedure

1. Expand Resources (or Environment) > IVRs > [specific IVR], right click IVR Ports, and select New > Range of IVR Ports. The Add a Range of IVR Ports dialog box appears.
2. In the Start Number box, enter the starting number for the range of IVR ports that you want to assign.
3. In the Number of IVR Ports box, enter the total number of ports that you want to assign.
4. Click OK to define the ports and list them in Configuration Manager.
[Table 12](#) shows the starting port number for the IVR types that the Genesys 7.5 IVR Drivers supports.

Table 12: Starting Number for IVR Ports

Genesys IVR Driver	Starting Port Number
Aspect Edify CONVERSANT Show N Tel	0
MPS500 & MPS1000 WVR for Windows	1

Table 12: Starting Number for IVR Ports (Continued)

Genesys IVR Driver	Starting Port Number
Microsoft Speech Server	<p>Six-digit number with the following syntax:</p> <ul style="list-style-type: none"> • Digit 1—The number that is assigned to the Telephony Application Services (TAS) instance • Digit 2—The board number • Digit 3—The trunk number • Digits 4–6—The port number, with leading zeroes <p>Example: 101001 is the complete port number for TAS instance 1, board number 0, trunk number 1, and port number 001 (port 1 with two leading zeroes).</p>
WVR for AIX	<p>WVR for AIX port numbers are derived using the following formula:</p> $(100 \times \text{T1/E1 number}) + 1$ <p>Examples: The first port on the first span is 101, and the first port on the second span is 201.</p>
GVP	<ul style="list-style-type: none"> • IVR-In-Front mode—The IP address as a character string (12 characters), followed by a three-digit port number (001–999). • IVR-Behind-Switch mode—The number starts with 01, and goes up to the number of ports.

5. Right-click a port number, and then select **Properties**. The Port Properties dialog box appears.
6. Click the **Folder** icon next to the **Associated DN** box to define the association between the IVR port and a DN for the switch that is connected to this IVR. The IVR port/DN association is completed through additional steps.
7. Repeat [Step 6](#) for each port in the range.

Note: If your IVR ports must be able to log in as agents, see [“Configuring the Auto-Login Feature.”](#)

End of procedure

Configuring the Auto-Login Feature

Procedure: Configuring the Auto-Login feature

Purpose: To configure the Auto-Login feature.

Start of procedure

1. Open the Port Properties dialog box, click the Annex tab.
2. Right-click in an empty area at the bottom of the Annex tab, and then select New. The Add Section dialog box appears.
3. In the Section Name box, enter AutoLogin.
4. Click OK. You are returned to the Port Properties dialog box, which now displays the new AutoLogin section on the Annex tab.
5. Double-click AutoLogin to open the AutoLogin section.
6. To define an AutoLogin option—for example, AgentID—right-click in an empty area at the bottom of the AutoLogin section, and then select New. The Edit Option dialog box appears.
7. In the Option Name and Option Value boxes, enter a name and value (respectively) for the new option.
8. Click OK. You are returned to the Port Properties dialog box, which now displays the new option in the AutoLogin section.
9. Repeat [Steps 5](#) and [6](#) to add more options.
10. When you have finished, click OK to close the Port Properties dialog box.

End of procedure

Configuring the I-Server Application

Note: The I-Server application is required only for the IVR-In-Front and IVR-Behind-Switch configurations. It is not used for the IVR Network T-Server configuration.

Procedure: Configuring the I-Server application

Purpose: To configure the I-Server application in Configuration Manager.

Start of procedure

1. Expand Environment, right-click Applications, and select New > Application. The Browse dialog box appears.
2. Select I-Server_810, and then click OK. The New I-Server_810 Properties dialog box appears, with the General tab displayed.
3. In the Name box, enter a unique name for the I-Server application.

Note: The name must not contain spaces.

4. Select the State Enabled check box to indicate that the object is in regular operating condition and can be used without restriction.
5. Click Apply to save your changes on this tab.
6. Click the Start Info tab.
7. Enter a single period (.) in the following boxes:
 - Working Directory
 - Command Line
 - Command Line Arguments
8. In the Startup and Shutdown boxes, accept the default values.
9. Make sure that the Auto-Restart check box is not selected.
10. Click Apply to save your changes on this tab.
11. Click the Server Info tab.
12. In the Host box, select the host on which the I-Server application is running.
13. In the Ports box, enter a number for the port. The port number should be set as “0” or “.” (or, in Configuration Manager 8.1, allow the value to be auto-suggested.)
14. Click Apply to save your changes on this tab.
15. Click the Connections tab.
16. Define a connection from the I-Server application to the TServer_IVR application, and, optionally, to the T-Server (for IVR-Behind-Switch mode only), Message Server (for centralized logging), and Stat Server. If you are configuring Hot Standby mode, this connection is in the primary IVR_Server_application connecting to the primary

TServer_IVR_application. The fact that you configure Hot Standby mode will cause this connection to also appear automatically in the backup IVR_Server_application.

Note: for hot standby deployments, the Primary TServer_IVR application should be entered in both the primary and backup I-Server application's Connections tab. For information about defining connections, see “Adding Connections for the I-Server Application” on [page 113](#).

17. Click Apply to save your changes on this tab.
18. Click the Options tab.
15. If you want to use Load Balancing, configure a LoadBalance section.
16. If you want to gather statistics, configure a stat:<stat name> section.
17. If you want to use routing, configure a VirtualRoutePoints section.

Note: For information about the options on the Options tab, see “I-Server Options” on [page 244](#).

18. Click Apply to save your changes on this tab.
19. Click OK to complete the configuration of the I-Server application.

End of procedure

Configuring the TServer_IVR Application

Notes:

- The TServer_IVR application is required only for the IVR-In-Front and IVR-Behind-Switch configurations. It is not used for the IVR Network T-Server configuration.
- Only the primary TServer_IVR application needs to be configured in the I-Server application Connection tab when in hot standby mode.

Procedure: Configuring the TServer_IVR application

Purpose: To configure the TServer_IVR application in Configuration Manager.

Start of procedure

1. Expand Environment, right-click Applications, and select New > Application. The Browse dialog box appears.
2. Select TServer_IVR_810, and then click OK. The New TServer_IVR_810 Properties dialog box appears, with the General tab displayed.
3. In the Name box, enter a unique name for the TServer_IVR application.

Note: The name must not contain spaces.

4. In the Tenant box, select the applicable tenant.

Note: The Tenant box is available only if you are in a multi-tenant environment.

5. In the Template box, select the applicable template.
6. Select the State Enabled check box to indicate that the object is in regular operating condition and can be used without restriction.
7. Click Apply to save your changes on this tab.
8. Click the Server Info tab.
9. In the Host box, select the host on which the TServer_IVR application is running.
10. In the Ports box, add the unique number of the port to which T-Library clients and IVR ports connect.
11. In the Backup Server box, select the server that should be contacted if the connection to the primary server fails.
12. In the Redundancy Type box, select the type of redundancy mode. Supported modes are Not Specified, Warm Standby, and Hot Standby.
13. In the Reconnect Timeout box, enter the amount of time (in seconds) that the client application waits between reconnection attempts after its connection with the server fails.
14. In the Reconnect Attempts box, enter the number of times that the client application attempts to connect to the primary server before it tries to connect to the backup server.

Note: This option is available only if a backup server is specified.

15. Click Apply to save your changes on this tab.
16. Click the Start Info tab.

17. In the **Working Directory** box, enter the full path to the directory in which this application is installed. The recommended installation directory path is:
 - For UNIX: `/home/gcti/TServer_IVR_810`
 - For Windows: `c:\gcti\TServer_IVR_810`
18. In the **Command Line** box, enter the command that is used to start this application:
 - For UNIX: `nts_server`
 - For Windows: `nts_server.exe`
19. In the **Command Line Arguments** box, enter values for the following command-line parameters:
 - `-host`: The host name of the Configuration Server.
 - `-port`: The port assigned to the Configuration Server.
 - `-app`: The name of the TServer_IVR application, as specified in [Step 3](#). If the name contains blank spaces, you must enclose it in double quotation marks.
20. In the **Startup and Shutdown** boxes, enter the time intervals (in seconds) during which this application is expected to start and shut down.
21. Make sure that the **Auto-Restart** check box is not selected.
22. Click **Apply** to save your changes on this tab.
23. Click the **Connections** tab..

Note: Connections to other T-Servers are required only if you have multi-site routing. For more information, see Chapter 3 on [page 47](#).

24. To add a server, see “Adding Servers” on [page 163](#).
25. Click **Apply** to save your changes on this tab.
26. Click the **Options** tab.
27. You must configure the following options:
 - `gli_server_address` in the `gli_server_group_1` section
 - `app_name` in the `IServer` section
 - `license-file` in the `License` section
 - `operation-mode` in the `IServerGLMSap` section

Note: In most cases, you can use the default values for the other options. For option descriptions, see “TServer_IVR Options” on [page 231](#).

To configure or edit the value for an option:

- a. Do one of the following:

- Double-click either the section name or option name.
- Click the Create New Section/Option button.

b. Enter the option name and option value.

c. Click OK.

If you want to create a new section, click the Create New Section/Option button, enter the section name, and click OK.

28. Click Apply to save your changes on this tab.
29. Click the Switches tab.
30. To add a virtual switch for the TServer_IVR application, click Add. The Browse dialog box appears.
31. Select a virtual switch, and then click OK. You are returned to the TServer_IVR_810 Properties dialog box, which now displays the virtual switch that you selected.
32. Click Apply to save your changes on this tab.
33. Click OK to complete the configuration of the TServer_IVR application.

End of procedure

Procedure:

Setting TServer_IVR application account permissions

Purpose: To set account permissions for the TServer_IVR application in Configuration Manager.

Start of procedure

1. Double-click the TServer_IVR icon. The TServer_IVR_810 Properties dialog box appears.
2. Click the Security tab.
3. Click Permissions to set permissions for the TServer_IVR application.
4. Select This Account, and then enter the login information.
5. Click OK.

End of procedure

Configuring the IVR_Driver Application

Note: The IVR_Driver application is required only for the IVR-In-Front and IVR-Behind-Switch configurations. However, it cannot be used with the IVR Drivers for Aspect and CONVERSANT. For information about how to configure these IVR Drivers, see the respective *System Administrator's Guide* for each one.

Procedure: Configuring the IVR_Driver application

Purpose: To configure the IVR_Driver application in Configuration Manager.

Start of procedure

1. Expand Environment, right-click Applications, and select New > Application. The Browse dialog box appears.
2. Select IVR_Driver_810, and then click OK. The New IVR_Driver_810 Properties dialog box appears, with the General tab displayed.
3. In the Name box, enter a unique name for the IVR_Driver application.

Note: The name must not contain spaces.

4. In the Template box, select the applicable template.
5. Select the State Enabled check box to indicate that the object is in regular operating condition and can be used without restriction.
6. Click Apply to save your changes on this tab.
7. Click the Server Info tab.
8. In the Host box, select the host on which the IVR_Driver application is running.
9. In the Ports box, add the unique number of the port to which T-Library clients connect.
10. In the Backup Server box, select the server that should be contacted if the connection to the primary server fails. Currently, None is the only supported value.
11. In the Redundancy Type box, select the type of redundancy mode. The only supported value for the IVR_Driver application is Not Specified.

12. In the Reconnect Timeout box, enter the amount of time (in seconds) that the client application waits between reconnection attempts after its connection with the server fails.
13. In the Reconnect Attempts box, enter the number of times that the client application attempts to connect to the primary server before it tries to connect to the backup server.

Note: This option is available only if a backup server is specified.

14. Click Apply to save your changes on this tab.
15. Click the Start Info tab.
16. Enter a single period (.) in the following boxes:
 - Working Directory
 - Command Line
 - Command Line Arguments
17. In the Startup and Shutdown boxes, enter time intervals (in seconds) during which this application is expected to start and shut down.
18. Make sure that the Auto-Restart check box is not selected.
19. Click Apply to save your changes on this tab.
20. Click the Connections tab.

Note: Only a connection to the Message Server is required for the IVR_Driver application.

21. To add a server, see “Adding Servers” on [page 163](#).
22. Click Apply to save your changes on this tab.
23. Click the Options tab.
24. Configure the load_sharing_servers options in the ivr_server_interface section.

Note: In most cases, you can use the default values for the other options. For option descriptions, see “IVR_Driver Options” on [page 248](#), and Chapter 11 on [page 209](#).

To configure or edit the value for an option:

- a. Do one of the following:
 - Double-click either the section name or option name.
 - Click the Create New Section/Option button.
- b. Enter the option name and option value.
- c. Click OK.

If you want to create a new section, click the **Create New Section/Option** button, enter the section name, and click **OK**.

25. Click **Apply** to save your changes on this tab.

26. Click **OK** to complete the configuration of the **IVR_Driver** application.

End of procedure

Procedure: **Setting IVR_Driver application account permissions**

Purpose: To set account permissions for the **IVR_Driver** application in Configuration Manager.

Start of procedure

1. Double-click the **IVR_Driver** icon. The **IVRDriver_810 Properties** dialog box appears.
2. Click the **Security** tab.
3. Click **Permissions** to set permissions for the **IVR_Driver** application.
4. Select **This Account**, and then enter the login information.
5. Click **OK**.

End of procedure

Configuring the TServer_IVR_Network Application

Note: The **TServer_IVR_Network** application is required only for the **IVR Network T-Server** configuration.

Procedure: **Configuring the TServer_IVR_Network application**

Purpose: To configure the **TServer_IVR_Network** application in Configuration Manager.

Start of procedure

1. Expand Environment, right-click Applications, and select New > Application. The Browse dialog box appears.
2. Select TServer_IVR_Network_810, and then click OK. The New TServer_IVR_Network_810 Properties dialog box appears, with the General tab displayed.
3. In the Name box, enter a unique name for the TServer_IVR_Network application.

Note: The name must not contain spaces.

4. In the Tenant box, select the applicable tenant.

Note: The Tenant box is available only if you are in a multi-tenant environment.

5. In the Template box, select the applicable template.
6. Select the State Enabled check box to indicate that the object is in regular operating condition and can be used without restriction.
7. Click Apply to save your changes on this tab.
8. Click the Server Info tab.
9. In the Host box, select the host on which the TServer_IVR_Network application is running.
10. In the Ports box, add the unique port number of the port to which T-Library clients connect.
11. In the Backup Server box, select the server that should be contacted if the connection to the primary server fails.
12. In the Redundancy Type box, select the type of redundancy mode. Supported modes are Not Specified, Warm Standby, and Hot Standby.
13. In the Reconnect Timeout box, enter the amount of time (in seconds) that the client application waits between reconnection attempts after its connection with the server fails.
14. In the Reconnect Attempts box, enter the number of times that the client application attempts to connect to the primary server before it tries to connect to the backup server.

Note: This option is available only if a backup server is specified.

15. Click Apply to save your changes on this tab.
16. Click the Start Info tab.

17. In the **Working Directory** box, enter the full path to the directory in which this application is installed. The recommended installation directory path is:
 - For UNIX: `/home/gcti/TServer_IVR_810`
 - For Windows: `c:\gcti\TServer_IVR_810`
18. In the **Command Line** box, enter the command that is used to start this application:
 - For UNIX: `nts_server`
 - For Windows: `nts_server.exe`
19. In the **Command Line Arguments** box, enter values for the following command-line parameters:
 - `-host`: The host name of the Configuration Server.
 - `-port`: The port assigned to the Configuration Server.
 - `-app`: The name of the TServer_IVR_Network application, as specified in [Step 3](#). If the name contains blank spaces, you must enclose it in double quotation marks.
20. In the **Startup and Shutdown** boxes, enter the time intervals (in seconds) during which this application is expected to start and shut down.
21. Make sure that the **Auto-Restart** check box is not selected.
22. Click **Apply** to save your changes on this tab.
23. Click the **Connections** tab.

Note: You must configure a connection to the Network T-Server application. Connections to the Message Server and a Premise T-Server are optional.

24. To add a server, see “Adding Servers” on [page 163](#).
25. Click **Apply** to save your changes on this tab.
26. Click the **Options** tab.
27. You must configure the following options:
 - `gli_server_address` in the `gli_server_group_1` section
 - `license-file` in the `license` section
 - `operation-mode` in the `IServerGLMSap` section

Note: In most cases, you can use the defaults for the other options. For option descriptions, see “TServer_IVR Options” on [page 231](#), and Chapter 11 on [page 209](#).

To configure or edit the value for an option:

- a. Do one of the following:

- Double-click either the section name or option name.
- Click the Create New Section/Option button.

b. Enter the option name and option value.

c. Click OK.

If you want to create a new section, click the Create New Section/Option button, enter the section name, and click OK.

28. Click Apply to save your changes on this tab.

29. Click the Switches tab.

30. To add a network switch for the TServer_IVR_Network application, click Add. The Browse dialog box appears.

31. Select a network switch, and then click OK. You are returned to the TServer_IVR_Network Properties dialog box, which now displays the network switch that you selected.

32. Click Apply to save your changes on this tab.

33. Click OK to complete the configuration of the TServer_IVR_Network application.

End of procedure

Procedure:

Setting TServer_IVR_Network application account permissions

Purpose: To set account permissions for the TServer_IVR_Network application in Configuration Manager.

Start of procedure

1. Double-click the TServer_IVR_Network icon. The TServer_IVR_Network_810 Properties dialog box appears.
2. Click the Security tab.
3. Click Permissions to set permissions for the TServer_IVR_Network application.
4. Select This Account, and then enter the login information.
5. Click OK.

End of procedure

Configuring Agent Logout for Load-Sharing IVR Servers

Starting with release 8.0, the Peer Support mode allows IVR Server to correctly log out agents even when multiple IVR Servers are being used in the Load-balanced mode.

Previously, there was a problem in that stopping any one of the IVR Servers would cause that IVR Server to log out agents even though remaining IVR Servers would be servicing the IVR clients. This made the maintenance of an IVR Server difficult in those cases where Load Balancing and IVR Server controlled agent login/logout was desired.

By configuring this feature, this problem can be avoided in this scenario. The group of IVR Servers will now send messages to each other so that any IVR Server that is commanded to stop can now conditionally logout agents if and only if there are no other running IVR Servers left in the group.

Configuring Peer Support

Use the Peer Support mode if you are using LoadBalance mode and are using IVR Server control of agent login/logout.

Procedure: Configuring Peer Support mode for IVR Server-controlled agent login/logout

Purpose: To enable IVR Server-controlled agent login/logout.

Start of procedure

1. Configure LoadBalance mode and agent login/logout as usual.
2. Configure the parameters [peer-list](#), [peer-mode-dn](#), and [peer-mode-timer](#) as desired.
3. On the virtual switch that is used with the load-balanced set of IVR Servers, configure the DN chosen in option `peer-mode-dn` as an extension.
4. In the I-Server application **Connection** tab of each of the IVR Servers, add a connection to each of the TServer_IVRs within the group.

End of procedure

Adding Servers

You can add servers on the **Connections** tab of the **New Application Properties** dialog box.

Procedure: Adding a server

Purpose: To add a server.

Start of procedure

1. Click **Add**. The **New Connection Info Properties** dialog box appears.
2. Click the **Folder** icon next to the **Server** box to display a list of server applications.
3. Select the applicable server, and then click **OK**.
4. In the **Connection Protocol** box, enter the connection control protocol.
5. In the **Local Timeout** box, enter the heartbeat polling interval on the client side; in the **Remote Timeout** box, enter the heartbeat polling interval on the server side.
6. In the **Trace Mode** box, select the trace mode.
7. Click **OK** to add the server.

End of procedure

8

High-Availability Deployment

This chapter describes how to set up a high-availability (HA) environment for your IVR Server. A high-availability architecture uses a pair of IVR Servers: a primary and a backup. These are monitored by a management application which, in a failure scenario, switches operations over to the other server without any significant loss of data.

The Framework Management Layer currently supports two types of configurations: warm standby and hot standby. In release 8.0 of IVR Server, hot standby high availability (*redundancy*) is implemented.

This chapter describes the high-availability architecture and how to configure IVR Server so that it operates with either type. Information in this chapter is divided into the following sections:

- [Warm Standby Redundancy Type, page 165](#)
- [Hot Standby Redundancy Type, page 167](#)
- [Prerequisites, page 169](#)
- [Warm Standby Deployment, page 170](#)
- [Hot Standby Deployment, page 173](#)
- [Limitations, page 175](#)

Warm Standby Redundancy Type

Genesys uses the expression *warm standby* to describe the redundancy type in which a backup server application remains initialized and ready to take over the operations of the primary server. The warm standby redundancy type reduces to a minimum the inability to process interactions that may have originated during the time it took to detect the failure. It also eliminates the need to bring a standby server online, thereby increasing solution availability.

Warm Standby Redundancy Architecture

Figure 14 illustrates the warm standby architecture. The standby server recognizes its role as a backup and does not process client requests until the Management Layer changes its role to primary. When a connection is broken between the primary server and the Local Control Agent (LCA, not shown in the diagram) running on the same host, a failure of the primary process is reported, and the switchover occurs; or, if the host on which the IVR Server is running fails, the switchover also occurs. (See the *Framework 8 Deployment Guide* for information on LCA.) As a result:

1. The Management Layer instructs the standby process to change its role from backup to primary.
2. A client application reconnects to the new primary.
3. The new primary (former backup) starts processing all new requests for service.

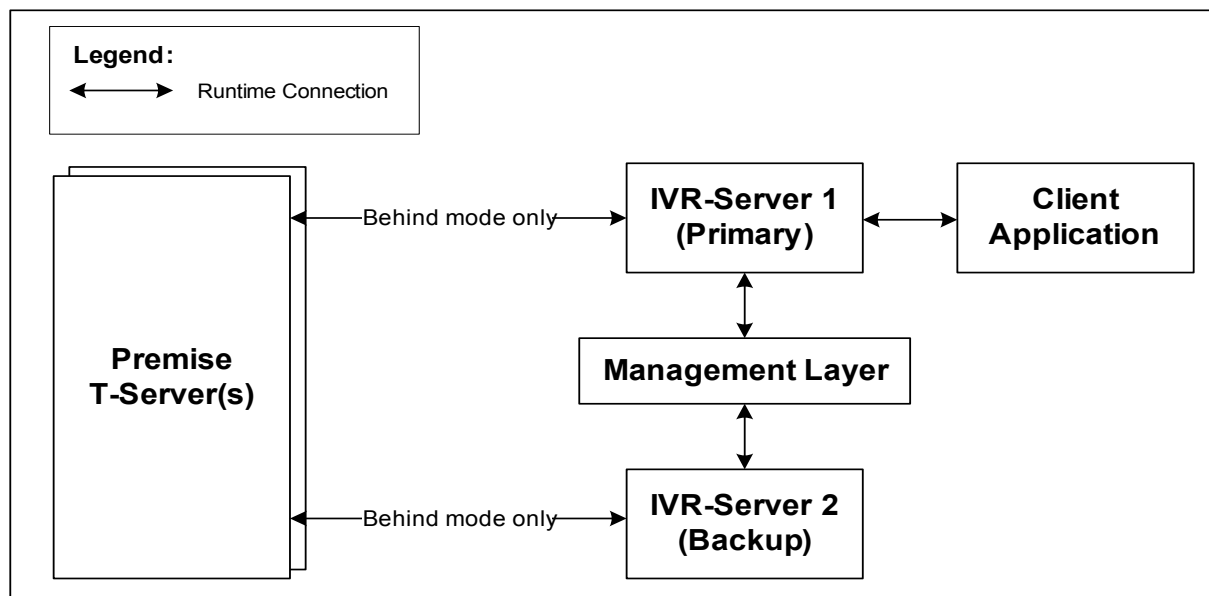


Figure 14: Warm Standby Redundancy Architecture

Note: In In-Front mode, IVR Server does not connect directly with a premise T-Server.

Although normal operations are restored as soon as the backup process takes over, the fault management effort continues. That effort consists of repeated

attempts to restart the process that failed. Once successfully restarted, the process is assigned the backup role.

Note: You can find full details on the role of the Management Layer in redundant configurations in the *Framework 8 Deployment Guide*.

Hot Standby Redundancy Type

Genesys uses the expression *hot standby* to describe the redundancy type in which a backup server application remains initialized, clients connect to both the primary and backup servers at startup, and the backup server data is synchronized from the primary server. Data synchronization and existing client connections to the backup guarantee higher availability of a component. (See Figure 15 on [page 168](#).)

Note: From T-Server release 7.1, the hot standby redundancy type is implemented in T-Servers for most types of switches. However, for some switches, you must compensate for the lack of link redundancy by using an additional Genesys component called *HA Proxy*.

Hot Standby Redundancy Architecture

Note: To use IVR Servers in Hot Standby mode, a separate additional license is needed. See the *Genesys Licensing Guide*.

[Figure 15](#) illustrates the switch-independent side of a hot standby implementation. Here, T-Servers start simultaneously and connect to the switch. At IVR startup, the Management Layer assigns the role of the primary server to IVR Server 1, and the role of backup to IVR Server 2.

IVR Server clients register with both IVR Servers, but only the primary IVR Server responds to client XML requests, except for the `LoginResp` and `FlowControl` (in other than Network mode). The internal IVR Server information, such as a DN status, `ConnID`, `UserData`, and `Call Type`, is synchronized between the primary and backup IVR Servers. Therefore, the backup IVR Server has the same information as the primary.

If IVR Server 1 fails, the Management Layer makes IVR Server 2 the new primary server, and it starts processing client requests. The Management Layer attempts to restart IVR Server 1, and if it is successful, it makes IVR Server 1 the new backup server.

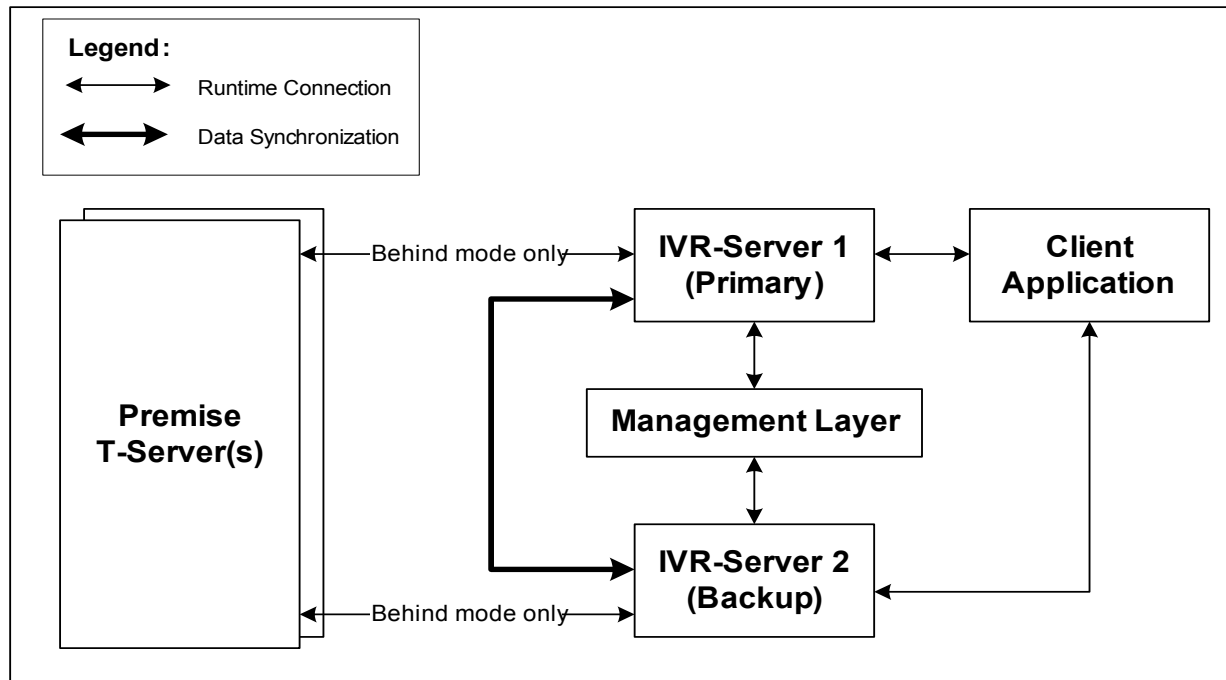


Figure 15: Hot Standby Redundancy Architecture

Note: In In-Front mode, IVR Server does not connect directly with a premise T-Server.

Benefits of Hot Standby Redundancy

The hot standby redundancy type provides the following benefits over the warm standby type:

- Using hot standby ensures the processing of interactions in progress if a failure occurs. After the primary IVR Server (IVR Server 1) fails, IVR Server 2 handles all new interactions and takes over the processing of interactions that are currently in progress.
- IVR Servers perform one-way (from primary to backup) synchronization of call-associated data, including, but not limited to:
 - Connection IDs.
 - Attached user data.

- Inter Server Call Control (ISCC; formerly called External Routing) call references to another site in a multi-site environment (to support the ISCC/COF feature).

Note: Refer to “ISCC Call Data Transfer Service” on [page 49](#) for ISCC feature descriptions.

- Allocation of ISCC-controlled resources.

However, keep the following hot standby limitations in mind:

- Client requests sent during the failure and switchover may be lost.
- Routing requests sent by the switch during the failure and switchover may be lost.
- IVR Server does not synchronize interactions that begin before it starts.
- Some T-Library events might be duplicated or lost.
- Reference IDs from IVR Server to a premise T-Server, in Behind mode, can be lost in events.

Prerequisites

This section presents basic requirements and recommendations for configuring and using T-Server high availability architectures.

Requirements

You must install the Management Layer if you are installing redundant IVR Server applications. In particular, install Local Control Agent (LCA) on each computer that runs IVR Server.

Warning! Genesys strongly recommends that you install the backup and primary IVR Servers on different host computers.

Synchronization Between IVR Server Pairs

When IVR Servers operate in a high-availability environment, the backup IVR Server must be ready to take on the primary role when required. For this purpose, both IVR Servers must be running and must have the same information.

When you configure IVR Server pairs to operate with the hot standby type, the primary IVR Server uses the connection to the backup to deliver synchronization updates. Genesys recommends that you enable the Advanced Disconnect Detection Protocol (ADDP), described in Chapter 2, for this connection. Do so using the configuration options in the

“Backup-Synchronization Section” section. Refer to the “T-Server Common Configuration Options” chapter for option descriptions.

Configuration Warnings

When configuring IVR Servers to support either the warm standby or hot standby redundancy type, remember:

1. When at least one of the two IVR Servers that operate in a redundant mode is running, do not change a redundancy type, host, or port in either IVR Server configuration.
2. When both the primary and backup IVR Servers are running, do not remove the backup IVR Server Application object from the configuration.

You must configure both IVR Servers in a high-availability pair to have the same options with the same values. If you change a value in one IVR Server configuration, you must change it in the other one manually.

The log options in the primary IVR Server can differ from those in the backup IVR Server configuration.

The link configuration options in the primary IVR Server can also differ from those in the backup IVR Server configuration.

Warm Standby Deployment

This section describes how to configure IVR Server pairs to work with the warm standby redundancy type, including details on their connections and settings.

General Order of Deployment

The general guidelines for IVR Server warm standby configuration are:

Manual Deployment

1. Manually configure two IVR Server Application objects.
2. Make sure the Switch object is configured for the switch these IVR Servers should serve.
3. Modify the configuration of the primary and backup IVR Servers as instructed in the following sections.

After completing the configuration steps, ensure that both IVR Servers are installed (see [page 172](#)).

Manual Modification of Two Independent IVR Servers to a Warm Standby Pair

Modify the configuration of both the primary and backup IVR Server Application objects as described in the following sections.

Note: From release 7.5, you can configure multiple ports for any application of type server. When multiple ports are configured for a server in a warm standby redundancy pair, the number of ports, their Port IDs, and the Listening Mode settings of the primary and backup servers must match respectively.

Procedure: Modifying the primary IVR Server configuration for warm standby

Start of procedure

1. Stop both the primary and backup IVR Servers if they are already running.
2. Open the Configuration Manager main window.
3. Open the Properties dialog box of the TServer_IVR_Application object for the IVR Server that you want to configure as a primary server.
4. Click the Switches tab.
5. Ensure that it specifies the Switch that this IVR Server Application should serve. If necessary, select the correct Switch using the Browse button.
6. Click Apply to save the configuration changes.
7. Click the Server Info tab.
8. Specify the TServer_IVR_Application object you want to use as the backup server. Use the Browse button next to the Backup Server field to locate the backup TServer_IVR_Application object.
9. Select Warm Standby as the Redundancy Type.
10. Click Apply to save the configuration changes.
11. Click the Start Info tab.
12. Select Auto-Restart.
13. Click Apply and OK to save the configuration changes.

End of procedure

Next Steps

- [Procedure: Modifying the backup IVR Server configuration for warm standby, on page 172](#)

Procedure:
Modifying the backup IVR Server configuration for warm standby

Note: For warm standby mode, the backup IVR Server must have load-balancing configuration values set in its `IVR_Server_Application` object, as described in this procedure.

Start of procedure

1. Open the Configuration Manager main window.
2. Open the Properties dialog box of the backup `TServer_IVR_Application` object for the IVR Server that you want to configure as a backup server.
3. Click the Switches tab.
4. Using the Browse button, select the same virtual Switch object you associated with the `TServer_IVR_Application` object in the primary IVR Server above.
5. Click Apply to save the configuration changes.
6. Click the Start Info tab.
7. Select Auto-Restart.
8. Click Apply and OK to save the configuration changes.
9. Open the Properties dialog box of the `IVR_Server_Application` object for the IVR Server that you want to configure as a backup server.
10. Click the Options tab.
11. Create a LoadBalance section.
12. In that section create variable `app-name` with a value of the name of the primary `IVRServer_Application` object.
13. Click Apply to save the configuration changes.

End of procedure

Warm Standby Installation of IVR Server Pairs

The installation of an IVR Server to be used in a high-availability pair is the same as that for the stand-alone IVR Server. If you have not installed the

primary and backup IVR Servers yet, follow the instructions for both installations.

Hot Standby Deployment

This section describes how to configure IVR Server pairs to work with the hot standby redundancy type, including details on their connections and settings.

General Order of Deployment

The general guidelines for IVR Server hot standby configuration are:

**Manual
Deployment**

1. Manually configure two IVR Server `Applications` objects as described in Chapter 7, “Manual Configuration,” on [page 135](#).
2. Make sure the `Switch` object is configured for the switch these IVR Servers should serve, as described in Chapter 7, “Manual Configuration,” on [page 135](#).
3. Modify the configuration of the primary and backup IVR Servers as instructed in the following sections.

After completing the configuration steps, ensure that both IVR Servers are installed (see [page 175](#)).

Manual Modification of Two Independent IVR Servers to a Hot Standby Pair

Modify the configuration of both the primary and backup IVR Server `Application` objects for hot standby redundancy as described in the following sections.

Note: From release 7.5, you can configure multiple ports for any application of type server. When multiple ports are configured for a server in a hot standby redundancy pair, the number of ports, their `Port IDs`, and the `Listening Mode` settings of the primary and backup servers must match respectively.

Procedure:

Modifying the primary IVR Server configuration for hot standby

Start of procedure

1. Open the Configuration Manager main window.
2. Open the Properties dialog box of the TServer_IVR_Application object for the IVR Server that you want to configure as a primary server.
3. Click the Switches tab.
4. Ensure that it specifies the Switch that this TServer_IVR_Application object should serve. If necessary, select the correct Switch using the Browse button.
5. Click Apply to save the configuration changes.
6. Click the Server Info tab.
7. Specify the TServer_IVR_Application object you want to use as the backup server. Use the Browse button next to the Backup Server field to locate the backup TServer_IVR_Application object.
8. Select Hot Standby as the Redundancy Type.
9. Click Apply to save the configuration changes.
10. Click the Start Info tab.
11. Select Auto-Restart.
12. Click Apply and OK to save the configuration changes.

End of procedure

Next Steps

- [Procedure: Modifying the backup IVR configuration for hot standby, on page 175](#)

Procedure: Modifying the backup IVR configuration for hot standby

Note: In Hot Standby the backup I-Server application must have the LoadBalance option configured, pointing to the primary I-Server.

Start of procedure

1. Open the Configuration Manager main window.
2. Open the Properties dialog box of the TServer_IVR_Application object for the IVR Server that you want to configure as a backup server.
3. Click the Switches tab.
4. Using the Browse button, select the same virtual Switch object you associated with the TServer_IVR_Application object in the primary IVR Server above.
5. Open the Properties dialog box of the IVR_Server_Application object for the IVR Server that you want to configure as a backup server.
6. Click the Options tab.
7. Create a LoadBalance section.
8. In that section create variable app-name with a value of the name of the primary IVRServer_Application.
9. Click Apply to save the configuration changes.

End of procedure

Hot Standby Installation of IVR Server Pairs

The installation of a IVR Server pairs is the same as that for stand-alone IVR Servers. If you have not installed the primary and backup IVR Servers yet, follow instructions in “Manual Configuration” on [page 135](#) for both installations.

Limitations

The IVR Server is not a true T-Server, since it can be running and fully connected to the Framework, but will not report “CTI Link Connected” till an IVR client connects. The premise T-Server connects to the switch CTI link, then reports status Started, even if no clients are connected. Because of this, manual mode switchover should only be attempted for IVR Server if the primary IVR Server is already in the Started state.

Please refer to the Solution Control Server chapter in the *Genesys Framework 8.1 Configuration Options Reference Manual*.

Note: Hot Standby high availability cannot be used alongside load balancing. The two modes are mutually exclusive.

9

Starting and Stopping IVR Server

This chapter describes how to start and stop IVR Server, which you can do only after you have properly installed and configured all the IVR Interface Option 8.1 components—for example, IVR Server and IVR Driver.

For information about installing and configuring IVR Server, see Chapter 5 on [page 127](#) and Chapter 7 on [page 135](#). For information about installing and configuring an IVR Driver, see the *IVR Interface Option 8.1 System Administrator's Guide* for your particular driver, or the documentation from the IVR Driver vendor.

This chapter contains the following sections:

- [Prestart Information, page 177](#)
- [Starting IVR Server, page 178](#)
- [Stopping IVR Server, page 179](#)
- [Starting and Stopping with Windows Services Manager, page 179](#)

Prestart Information

Genesys recommends the following:

- Use the Management Layer (the Solution Control Interface [SCI]) to start and stop IVR Server.
- If you are using network logging, start the Message Server before IVR Server.
- Start IVR Server before IVR Driver.

Starting IVR Server

After you install and configure the IVR Interface Option 8.1 components, you can start IVR Server on either UNIX or Windows operating systems.

Note: Although you must configure two applications for IVR Server: I-Server and TServer_IVR, you need to start and stop only TServer_IVR, not I-Server.

UNIX

To start IVR Server on UNIX, enter one of the following commands at the command line:

```
run.sh  
nts_server -host host_name -port port_number -app app_name -nco X/Y
```

In the second command, the command-line parameters are used as follows:

-host	The value following this key is the actual host name.
-port	The value following this key is the actual port name.
-app	The value following this key is the actual application name.
-nco X/Y	Enables the Nonstop Operation feature: X exceptions occurring within Y seconds do not cause an application to close. If the specified number of exceptions is exceeded within the specified number of seconds, the application closes or, if it is configured to do so, the Management Layer restarts the application. If the -nco parameter is not specified, the default value of 6 exceptions handled in 10 seconds is used. To disable the Nonstop Operation feature, specify -nco 0 in the start command.

Note: You can also add the license file name to the startup command, by using the -l parameter. However, Genesys recommends that you enter license information on the Options tab of the TServer_IVR application's Properties dialog box.

Windows

To start IVR Server on Windows, enter the following command at the command line:

```
nts_server -host host_name -port port_number -app app_name
```

The command-line parameters are used as follows:

-host	The value following this key is the actual host name.
-port	The value following this key is the actual port name.

`-app` The value following this key is the actual application name.

Note: You can also add the license file name to the startup command, by using the `-l` parameter. However, Genesys recommends that you enter license information on the Options tab of the TServer_IVR application's Properties dialog box.

Stopping IVR Server

After you start the IVR Interface Option 8.1 components, you can stop IVR Server on either UNIX or Windows operating systems.

UNIX

To stop IVR Server on UNIX, enter the following command at the command line:

```
ps -ef | grep nts_server
kill -9 <process_id>
```

Windows

To stop IVR Server on Windows:

1. Open the Task Manager.
2. Select the `nts_server.exe` process.
3. Click End Task.

Starting and Stopping with Windows Services Manager

When you start an application that is installed as a Windows Service, make sure that you correctly specify the startup parameters of the application in the ImagePath, in the Application folder in the Registry Editor. The ImagePath must have the following value data:

```
<full path>\<executable file name> -service <Application Name as Service> -host <Configuration Server host>
-port <Configuration Server port> -app <Application Name>
-l <license address>
```

For the command-line parameters that are common to Framework server components, see [page 178](#) and the following:

`-service` The value following this key is the name of the application that is running as a Windows Service; typically, it matches the

application name that is specified in the `-app` command-line parameter.

Framework components that are installed as Windows Services with the autostart capability are automatically started each time the computer on which they are installed is restarted.

To start Framework components that are installed as Windows Services with the manual start capability, click **Start** in **Services Manager**.

Regardless of a component's start mode, you can stop Framework components that are installed as Windows Services by clicking **Stop** in **Services Manager**.

Note: Use the **Windows Services** dialog box to change the startup mode from **Automatic** to **Manual**, and vice versa.



Chapter

10

T-Server Common Configuration Options

This chapter describes the configuration options that are generally common to all T-Server types, with some exceptions noted. It contains the following sections:

- [Setting Configuration Options, page 181](#)
- [Mandatory Options, page 182](#)
- [TServer Section, page 182](#)
- [license Section, page 187](#)
- [agent-reservation Section, page 190](#)
- [extrouter Section, page 191](#)
- [backup-sync Section, page 202](#)
- [call-cleanup Section, page 204](#)
- [Translation Rules Section, page 205](#)
- [security Section, page 206](#)
- [Timeout Value Format, page 206](#)
- [Changes from Release 8.0 to 8.1, page 207](#)

T-Server also supports common log options described in Chapter 11, “Common Configuration Options,” on [page 209](#).

Setting Configuration Options

Unless specified otherwise, set T-Server common configuration options in the Options of the Application object, using one of the following navigation paths:

- In Genesys Administrator—Application object > Options tab > Advanced View (Options)
- In Configuration Manager—Application object > Properties dialog box > Options tab

Mandatory Options

Except as noted for certain environments, the configuration of common options is not required for basic T-Server operation.

TServer Section

The TServer section contains the configuration options that are used to support the core features common to all T-Servers.

This section must be called TServer.

ani-distribution

Default Value: inbound-calls-only

Valid Values: inbound-calls-only, all-calls, suppressed

Changes Take Effect: Immediately

Controls the distribution of the ANI information in TEvent messages. When this option is set to all-calls, the ANI attribute will be reported for all calls for which it is available. When this option is set to suppressed, the ANI attribute will not be reported for any calls. When this option is set to inbound-calls-only, the ANI attribute will be reported for inbound calls only.

background-processing

Default Value: true

Valid Values: true, false

Changes Take Effect: Immediately

When set to true, T-Server processes all client requests in the background, giving higher priority to the rest of the messages. This ensures that it processes these messages without any significant delay.

With Background Processing functionality enabled, T-Server processes all switch messages immediately and waits until there are no switch messages before processing the message queue associated with T-Server client requests. T-Server reads all connection sockets immediately and places client requests in the input buffer, which prevents T-Server clients from disconnecting because of configured timeouts.

When T-Server processes client requests from the message queue, requests are processed in the order in which T-Server received them.

When set to false, T-Server processes multiple requests from one T-Server client before proceeding to the requests from another T-Server client, and so on.

background-timeout

Default Value: 60 msec

Valid Values: See “Timeout Value Format” on [page 206](#).

Changes Take Effect: Immediately

Specifies the time interval that T-Server waits before processing client requests in background mode. You must set the `background-processing` option to `true` in order for this option to take effect.

check-tenant-profile

Default Value: `false`

Valid Values: `true`, `false`

Changes Take Effect: For the next connected client

When set to `true`, T-Server only allows a client to register if the client provides the correct name and password of a T-Server Tenant. If the client provides the Tenant name concatenated with a slash (/) and the Tenant password for the Tenant to which T-Server belongs as the value of `AttributeApplicationPassword` in the `TRegisterClient` request, T-Server allows that client to register DNs that are included in the switch configuration in the Configuration Database, but it does not allow the client to register DNs that are *not* included in the switch configuration.

consult-user-data

Default Value: `separate`

Valid Values:

<code>separate</code>	Stores user data for original and consultation calls in separate structures. The data attached to the original call is available for review or changes only to the parties of that call. The data attached to the consultation call is available only to the parties of the consultation call.
<code>inherited</code>	Copies user data from an original call to a consultation call when the consultation call is created; thereafter, stores user data separately for the original and the consultation call. Changes to the original call's user data are not available to the parties of the consultation call, and vice versa.
<code>joint</code>	Stores user data for an original call and a consultation call in one structure. The user data structure is associated with the original call, but the parties of both the original and consultation calls can see and make changes to the common user data.

Changes Take Effect: For the next consultation call created

Specifies the method for handling user data in a consultation call.

Note: A T-Server client can also specify the `consult-user-data` mode in the `Extensions` attribute `ConsultUserData` key for a conference or transfer request. If it is specified, the method of handling user data is based on the value of the `ConsultUserData` key-value pair of the request and takes precedence over the T-Server `consult-user-data` option. If it is not specified in the client request, the value specified in the `consult-user-data` option applies.

customer-id

Default Value: No default value. (A value must be specified for a multi-tenant environment.)

Valid Values: Any character string

Changes Take Effect: Immediately

Identifies the T-Server customer. You must set this option to the name of the tenant that is using this T-Server. You must specify a value for this option if you are working in a multi-tenant environment.

Note: Do not configure the `customer-id` option for single-tenant environments.

dn-scope

Default Value: `undefined`

Valid Values: `undefined`, `switch`, `office`, `tenant`

Changes Take Effect: Immediately

Related Feature: “Switch Partitioning” on [page 86](#)

Specifies whether DNs associated with the `Switch`, `Switching Office`, or `Tenant` objects will be considered in the T-Server monitoring scope, enabling T-Server to report calls to or from those DNs as internal.

With a value of `tenant`, all DNs associated with the switches that are within the `Tenant` will be in the T-Server monitoring scope. With a value of `office`, all DNs associated with the switches that are within the `Switching Office` will be in the T-Server monitoring scope. With a value of `switch`, all DNs associated with the `Switch` will be in the T-Server monitoring scope.

With a value of `undefined` (the default), pre-8.x T-Server behavior applies and the switch partitioning is not turned on.

Note: Setting the option to a value of `office` or `tenant`, which requires T-Server to monitor a large set of configuration data, may negatively affect T-Server performance.

log-trace-flags

Default Value: `+iscc, +cfg$dn, -cfgserv, +passwd, +udata, -devlink, -sw, -req, -callops, -conn, -client`

Valid Values (in any combination):

<code>+/-iscc</code>	Turns on/off the writing of information about Inter Server Call Control (ISCC) transactions.
<code>+/-cfg\$dn</code>	Turns on/off the writing of information about DN configuration.
<code>+/-cfgserv</code>	Turns on/off the writing of messages from Configuration Server.
<code>+/-passwd</code>	Turns on/off the writing of <code>AttributePassword</code> in <code>TEvents</code> .
<code>+/-udata</code>	Turns on/off the writing of attached data.
<code>+/-devlink</code>	Turns on/off the writing of information about the link used to send CTI messages to the switch (for multilink environments).
<code>+/-sw</code>	Reserved by Genesys Engineering.
<code>+/-req</code>	Reserved by Genesys Engineering.
<code>+/-callops</code>	Reserved by Genesys Engineering.
<code>+/-conn</code>	Reserved by Genesys Engineering.
<code>+/-client</code>	Turns on/off the writing of additional information about the client's connection.

Changes Take Effect: Immediately

Specifies—using a space-, comma- or semicolon-separated list—the types of information that are written to the log files.

management-port

Default Value: `0`

Valid Values: `0` or any valid TCP/IP port

Changes Take Effect: After T-Server is restarted

Specifies the TCP/IP port that management agents use to communicate with T-Server. If set to `0` (zero), this port is not used.

merged-user-data

Default Value: `main-only`

Valid Values:

<code>main-only</code>	T-Server attaches user data from the remaining call only.
<code>merged-only</code>	T-Server attaches user data from the merging call.
<code>merged-over-main</code>	T-Server attaches user data from the remaining and the merging call. In the event of equal keys, T-Server uses data from the merging call.
<code>main-over-merged</code>	T-Server attaches data from the remaining and the merging call. In the event of equal keys, T-Server uses data from the remaining call.

Changes Take Effect: Immediately

Specifies the data that is attached to the resulting call after a call transfer, conference, or merge completion.

Note: The option setting does not affect the resulting data for merging calls if the `consult-user-data` option is set to `joint`. (See “consult-user-data” on [page 183](#).)

propagated-call-type

Default Value: `false`

Valid Values: `true`, `false`

Changes Take Effect: Immediately

Related Feature: “Switch Partitioning” on [page 86](#)

Determines what T-Server reports as the value of the `CallType` attribute in events related to calls that have been synchronized with another site via ISCC, as follows:

- When set to `false`, T-Server reports in events related to calls that have been synchronized with another site via ISCC the same value for the `CallType` attribute as it did in pre-8.0 releases and adds the new `PropagatedCallType` attribute with the value of the `CallType` attribute at the origination site. This provides backward compatibility with existing T-Server clients.
- When set to `true`, T-Server reports in events related to calls that have been synchronized with another site via ISCC the same value for the `CallType` attribute as at the origination site, and adds the new `LocalCallType` attribute with the same value as `CallType` in pre-8.0 releases.

server-id

Default Value: An integer equal to the value `ApplicationDBID` as reported by Configuration Server

Valid Values: Any integer from 0–16383

Changes Take Effect: Immediately

Specifies the `Server ID` that T-Server uses to generate `Connection IDs` and other unique identifiers. In a multi-site environment, you must assign each T-Server a unique `Server ID`, in order to avoid confusion in reporting applications and T-Server behavior.

Configuration of this option is necessary for Framework environments in which there are two or more instances of the Configuration Database.

Note: If you do not specify a value for this option, T-Server populates it with the `ApplicationDBID` as reported by Configuration Server. Each data object in the Configuration Database is assigned a separate DBID that maintains a unique `Server ID` for each T-Server configured in the database.

Warning! Genesys does not recommend using multiple instances of the Configuration Database.

user-data-limit

Default Value: 16000

Valid Values: 0–65535

Changes Take Effect: Immediately

Specifies the maximum size (in bytes) of user data in a packed format.

Note: When T-Server works in mixed 8.x/7.x/6.x environment, the value of this option must not exceed the default value of 16000 bytes; otherwise, 6.x T-Server clients might fail.

license Section

The License section contains the configuration options that are used to configure T-Server licenses. They set the upper limit of the seat-related DN licenses (`tserver_sdn`) that T-Server tries to check out from a license file. See “License Checkout” on [page 188](#).

This section must be called `license`.

Notes:

- T-Server also supports the `license-file` option described in the *Genesys Licensing Guide*.
- The `license` section is not applicable to Network T-Server for DTAG.
- On selected platforms, the limitation of 9999 licenses may no longer apply. Use values greater than 9999 only when instructed by Genesys Technical Support.

If you use two or more T-Servers, and they share licenses, you must configure the following options in the `license` section of the T-Servers.

num-of-licenses

Default Value: 0 or `max` (all available licenses)

Valid Values: String `max` or any integer from 0–9999

Changes Take Effect: Immediately

Specifies how many DN licenses T-Server checks out. T-Server treats a value of 0 (zero) the same as it treats `max`—that is, it checks out all available licenses.

The sum of all `num-of-licenses` values for all concurrently deployed T-Servers must not exceed the number of seat-related DN licenses (`tserver_sdn`) in the corresponding license file. The primary and backup

T-Servers share the same licenses, and therefore they need to be counted only once. T-Server checks out the number of licenses indicated by the value for this option, regardless of the number actually in use.

num-sdn-licenses

Default Value: 0 or max (all DN licenses are seat-related)

Valid Values: String max (equal to the value of num-of-licenses), or any integer from 0–9999

Changes Take Effect: Immediately

Specifies how many seat-related licenses T-Server checks out. A value of 0 (zero) means that T-Server does not grant control of seat-related DNs to any client, and it does not look for seat-related DN licenses at all.

The sum of all num-sdn-licenses values for all concurrently deployed T-Servers must not exceed the number of seat-related DN licenses (tserver_sdn) in the corresponding license file. The primary and backup T-Servers share the same licenses, and therefore they need to be counted only once. T-Server checks out the number of licenses indicated by the value for this option, regardless of the number actually in use.

-
- Notes:**
- For Network T-Servers, Genesys recommends setting this option to 0.
 - Be sure to configure in the Configuration Database all the DNs that agents use (Extensions and ACD Positions) and that T-Server should control.
-

License Checkout

Table 13 shows how to determine the number of seat-related DN licenses that T-Server attempts to check out. See the examples on [page 189](#).

Table 13: License Checkout Rules

Options Settings ¹		License Checkout ²
num-of-licenses	num-sdn-licenses	Seat-related DN licenses
max (or 0)	max	all available
max (or 0)	x	x
max (or 0)	0	0
x	max	x

Table 13: License Checkout Rules (Continued)

Options Settings ¹		License Checkout ²
num-of-licenses	num-sdn-licenses	Seat-related DN licenses
x	y	min (y, x)
x	0	0

1. In this table, the following conventions are used: x and y - are positive integers; max is the maximum number of licenses that T-Server can check out; min (y, x) is the lesser of the two values defined by y and x, respectively.
2. The License Checkout column shows the number of licenses that T-Server attempts to check out. The actual number of licenses will depend on the licenses' availability at the time of checkout, and it is limited to 9999.

Examples

This section presents examples of option settings in the license section.

Example 1

If...		Then...
Options Settings	License File Settings	License Checkout
num-of-licenses = max	tserver_sdn = 500	500 seat-related DNs
num-sdn-licenses = max		

Example 2

If...		Then...
Options Settings	License File Settings	License Checkout
num-of-licenses = 1000	tserver_sdn = 500	500 seat-related DNs
num-sdn-licenses = max		

Example 3

If...		Then...
Options Settings	License File Settings	License Checkout
num-of-licenses = 1000	tserver_sdn = 600	400 seat-related DN's
num-sdn-licenses = 400		

Example 4

If...		Then...
Options Settings	License File Settings	License Checkout
num-of-licenses = max	tserver_sdn = 5000	1000 seat-related DN's
num-sdn-licenses = 1000		

agent-reservation Section

The `agent-reservation` section contains the configuration options that are used to customize the T-Server Agent Reservation feature. See “Agent Reservation” on [page 44](#) section for details on this feature.

This section must be called `agent-reservation`.

Note: The Agent Reservation functionality is currently a software-only feature that is used to coordinate multiple client applications. This feature does not apply to multiple direct or ACD-distributed calls.

collect-lower-priority-requests

Default Value: `true`

Valid Values: `true`, `false`

Changes Take Effect: Immediately

Specifies whether an agent reservation request is collected, depending on its priority during the time interval specified by the `request-collection-time` configuration option. When set to `false`, during the `request-collection-time` interval T-Server collects reservation requests of the highest priority only, rejecting newly submitted requests that have a lower priority or rejecting all previously submitted requests if a request with a higher priority arrives. When set to `true` (the default), agent reservation requests are collected as they were in pre-8.x releases.

reject-subsequent-request

Default Value: `true`

Valid Values:

- | | |
|--------------------|---|
| <code>true</code> | T-Server rejects subsequent requests. |
| <code>false</code> | A subsequent request prolongs the current reservation made by the same client application for the same agent. |

Changes Take Effect: Immediately

Specifies whether T-Server rejects subsequent requests from the same client application, for an agent reservation for the same Agent object that is currently reserved.

Note: Genesys does not recommend setting this option to `false` in a multi-site environment in which remote locations use the Agent-Reservation feature.

request-collection-time

Default Value: `100 msec`

Valid Values: See “Timeout Value Format” on [page 206](#).

Changes Take Effect: Immediately

Specifies the interval that agent reservation requests are collected before a reservation is granted. During this interval, agent reservation requests are delayed, in order to balance successful reservations between client applications (for example, Universal Routing Servers).

reservation-time

Default Value: `10000 msec`

Valid Values: See “Timeout Value Format” on [page 206](#).

Changes Take Effect: Immediately

Specifies the default interval for which an Agent DN is reserved. During this interval, the agent cannot be reserved again.

extrouter Section

The `extrouter` section contains the configuration options that are used to support multi-site environments with the Inter Server Call Control (ISCC) feature. The configuration options in this section of the document are grouped with related options that support the same functionality, as follows:

- [ISCC Transaction Options, page 193](#)
- [Transfer Connect Service Options, page 197](#)
- [ISCC/COF Options, page 198](#)
- [Event Propagation Options, page 200](#)
- [Number Translation Option, page 201](#)
- [GVP Integration Option, page 202](#)

This configuration section must be called `extrouter`.

For a description of the ways in which T-Server supports multi-site configurations and for an explanation of the configuration possibilities for a multi-site operation, see the “[Multi-Site Support](#)” chapter.

Note: In a multi-site environment, you must configure the `timeout`, `cast-type`, and `default-dn` options with the same value for both the primary and backup T-Servers. If you do not do this, the value specified for the backup T-Server overrides the value specified for the primary T-Server.

match-call-once

Default Value: `true`

Valid Values:

- | | |
|--------------------|--|
| <code>true</code> | ISCC does not process (match) an inbound call that has already been processed (matched). |
| <code>false</code> | ISCC processes (attempts to match) a call as many times as it arrives at an ISCC resource or multi-site-transfer target. |

Changes Take Effect: Immediately

Specifies how many times ISCC processes an inbound call when it arrives at an ISCC resource. When set to `false`, ISCC processes (attempts to match) the call even if it has already been processed.

Note: Genesys does not recommend changing the default value of the `match-call-once` option to `false` unless you have specific reasons. Setting this option to `false` may lead to excessive or inconsistent call data updates.

reconnect-tout

Default Value: `5 sec`

Valid Values: See “Timeout Value Format” on [page 206](#).

Changes Take Effect: At the next reconnection attempt

Specifies the time interval after which a remote T-Server attempts to connect to this T-Server after an unsuccessful attempt or a lost connection. The number of attempts is unlimited. At startup, T-Server immediately attempts the first connection, without this timeout.

report-connid-changes

Default Value: `false`

Valid Values:

- | | |
|--------------------|--|
| <code>true</code> | <code>EventPartyChanged</code> is generated. |
| <code>false</code> | <code>EventPartyChanged</code> is not generated. |

Changes Take Effect: Immediately

Specifies whether the destination T-Server generates `EventPartyChanged` for the incoming call when the resulting `ConnID` attribute is different from the `ConnID` attribute of an instance of the same call at the origination location.

use-data-from

Default Value: `current`

Valid Values:

<code>active</code>	The values of <code>UserData</code> and <code>ConnID</code> attributes are taken from the consultation call.
<code>original</code>	The values of <code>UserData</code> and <code>ConnID</code> attributes are taken from the original call.
<code>active-data-original-call</code>	The value of the <code>UserData</code> attribute is taken from the consultation call and the value of <code>ConnID</code> attribute is taken from the original call.
<code>current</code>	<p>If the value of <code>current</code> is specified, the following occurs:</p> <ul style="list-style-type: none"> • Before the transfer or conference is completed, the <code>UserData</code> and <code>ConnID</code> attributes are taken from the consultation call. • After the transfer or conference is completed, <code>EventPartyChanged</code> is generated, and the <code>UserData</code> and <code>ConnID</code> are taken from the original call.

Changes Take Effect: Immediately

Specifies the call from which the values for the `UserData` and `ConnID` attributes are taken for a consultation call that is routed or transferred to a remote location.

Note: For compatibility with the previous T-Server releases, you can use the values `consult`, `main`, and `consult-user-data` for this option. These are aliases for `active`, `original`, and `current`, respectively.

ISCC Transaction Options

cast-type

Default Value: `route`, `route-uu`, `reroute`, `direct-callid`, `direct-uu`, `direct-network-callid`, `direct-notoken`, `direct-digits`, `direct-ani`, `dnis-pool`, `pullback`

Valid Values: `route`, `route-uu`, `reroute`, `direct-callid`, `direct-uu`, `direct-network-callid`, `direct-notoken`, `direct-digits`, `direct-ani`, `dnis-pool`, `pullback`

Changes Take Effect: For the next request for the remote service

Specifies—using a space-, comma- or semicolon-separated list—the routing types that can be performed for this T-Server.

The valid values provide for a range of mechanisms that the ISCC feature can support with various T-Servers, in order to pass call data along with calls between locations.

Because switches of different types provide calls with different sets of information parameters, some values might not work with your T-Server. See Table 3 on [page 65](#) for information about supported transaction types by a specific T-Server. The “[Multi-Site Support](#)” chapter also provides detailed descriptions of all transaction types.

Notes: For compatibility with the previous T-Server releases, you can use the `direct` value for this option. This is an alias for `direct-callid`.

An alias, `route-notoken`, has been added to the `route` value.

default-dn

Default Value: No default value

Valid Values: Any DN

Changes Take Effect: For the next request for the remote service

Specifies the DN to which a call is routed when a Destination DN (`AttributeOtherDN`) is not specified in the client’s request for routing. If neither this option nor the client’s request contains the destination DN, the client receives `EventError`.

Note: This option is used only for requests with route types `route`, `route-uui`, `direct-callid`, `direct-network-callid`, `direct-uui`, `direct-notoken`, `direct-digits`, and `direct-ani`.

direct-digits-key

Default Value: `CDT_Track_Num`

Valid Values: Any valid key name of a key-value pair from the `UserData` attribute

Changes Take Effect: For the next request for the remote service

Specifies the name of a key from the `UserData` attribute that contains a string of digits that are used as matching criteria for remote service requests with the `direct-digits` routing type.

Note: For compatibility with the previous T-Server releases, this configuration option has an alias value of `cdt-udata-key`.

dn-for-unexpected-calls

Default Value: No default value

Valid Values: Any DN

Changes Take Effect: Immediately

Specifies a default DN for unexpected calls arriving on an External Routing Point.

network-request-timeout

Default Value: 20 sec

Valid Values: See “Timeout Value Format” on [page 206](#).

Changes Take Effect: For the next network request

For a premise T-Server, this option specifies the time interval that the premise T-Server waits for a response, after relaying a TNetwork<...> request to the Network T-Server. For a Network T-Server, this option specifies the time interval that the Network T-Server waits for a response from an SCP (Service Control Point), after initiating the processing of the request by the SCP.

When the allowed time expires, the T-Server cancels further processing of the request and generates EventError.

register-attempts

Default Value: 5

Valid Values: Any positive integer

Changes Take Effect: For the next registration

Specifies the number of attempts that T-Server makes to register a dedicated External Routing Point.

register-tout

Default Value: 2 sec

Valid Values: See “Timeout Value Format” on [page 206](#).

Changes Take Effect: For the next registration

Specifies the time interval after which T-Server attempts to register a dedicated External Routing Point. Counting starts when the attempt to register a Routing Point fails.

request-tout

Default Value: 20 sec

Valid Values: See “Timeout Value Format” on [page 206](#).

Changes Take Effect: For the next request for remote service

Specifies the time interval that a T-Server at the origination location waits for a notification of routing service availability from the destination location. Counting starts when the T-Server sends a request for remote service to the destination site.

resource-allocation-modeDefault Value: `circular`

Valid Values:

`home` T-Server takes an alphabetized (or numerically sequential) list of configured DN's and reserves the first available DN from the top of the list for each new request. For example, if the first DN is not available, the second DN is allocated for a new request. If the first DN is freed by the time the next request comes, the first DN is allocated for this next request.

`circular` T-Server takes the same list of configured DN's, but reserves a subsequent DN for each subsequent request. For example, when the first request comes, T-Server allocates the first DN; when the second request comes, T-Server allocates the second DN; and so on. T-Server does not reuse the first DN until reaching the end of the DN list.

Changes Take Effect: Immediately

Specifies the manner in which T-Server allocates resources (that is, DN's of the External Routing Point type and Access Resources with the Resource Type set to `dnis`) for multi-site transaction requests.

resource-load-maximumDefault Value: `0`

Valid Values: Any positive integer

Changes Take Effect: Immediately

Specifies the maximum number of ISCC routing transactions that can be concurrently processed at a single DN of the External Routing Point route type. After a number of outstanding transactions at a particular DN of the External Routing Point type reaches the specified number, T-Server considers the DN not available. Any subsequent request for this DN is queued until the number of outstanding transactions decreases. A value of `0` (zero) means that no limitation is set to the number of concurrent transactions at a single External Routing Point. In addition, the `0` value enables T-Server to perform load balancing of all incoming requests among all available External Routing Points, in order to minimize the load on each DN.

route-dn

Default Value: No default value

Valid Values: Any DN

Changes Take Effect: Immediately

Specifies the DN that serves as a Routing Point for the `route` transaction type in the multiple-to-one access mode.

timeout

Default Value: 60 sec

Valid Values: See “Timeout Value Format” on [page 206](#).

Changes Take Effect: For the next request for remote service

Specifies the time interval that the destination T-Server waits for a call routed from the origination location. Counting starts when this T-Server notifies the requesting T-Server about routing service availability. The timeout must be long enough to account for possible network delays in call arrival.

use-implicit-access-numbers

Default Value: false

Valid Values: true, false

Changes Take Effect: After T-Server is restarted

Determines whether an External Routing Point in which at least one access number is specified is eligible for use as a resource for calls coming from switches for which an access number is not specified in the External Routing Point. If this option is set to false, the External Routing Point is not eligible for use as a resource for calls coming from such switches. If this option is set to true, an implicit access number for the External Routing Point, composed of the switch access code and the DN number of the External Routing Point, will be used.

Note: If an External Routing Point does not have an access number specified, this option will not affect its use.

Transfer Connect Service Options

tcs-queue

Default Value: No default value

Valid Values: Any valid DN number

Changes Take Effect: For the next request for the remote service

Specifies the TCS DN number to which a call, processed by the TCS feature, is dialed after the originating external router obtains an access number. This option applies only if the [tcs-use](#) option is activated.

tcs-use

Default Value: never

Valid Values:

never	The TCS feature is not used.
always	The TCS feature is used for every call.
app-defined	In order to use the TCS feature for a multi-site call transfer request, a client application must add a key-value pair with a TC-type key and a nonempty string value to the UserData attribute of the request.

Changes Take Effect: Immediately

Specifies whether the Transfer Connect Service (TCS) feature is used.

Note: For compatibility with the previous T-Server releases, you can use the value `up-app-depended` for this option. This is an alias for `app-defined`.

ISCC/COF Options

cof-ci-defer-create

Default Value: 0

Valid Values: See “Timeout Value Format” on [page 206](#).

Changes Take Effect: Immediately

Specifies the time interval that T-Server waits for call data from the switch before generating a negative response for a call data request from a remote T-Server. If T-Server detects the matching call before this timeout expires, it sends the requested data. This option applies only if the `cof-feature` option is set to true.

cof-ci-defer-delete

Default Value: 0

Valid Values: See “Timeout Value Format” on [page 206](#).

Changes Take Effect: Immediately

Specifies the time interval that T-Server waits before deleting call data that might be overflowed. If set to 0, deletion deferring is disabled. This option applies only if the `cof-feature` option is set to true.

cof-ci-req-tout

Default Value: 500 msec

Valid Values: See “Timeout Value Format” on [page 206](#).

Changes Take Effect: For the next COF operation

Specifies the time interval during which T-Server will wait for call data requested with respect to a call originated at another site. After T-Server sends the call data request to remote T-Servers, all events related to this call will be

suspended until either the requested call data is received or the specified timeout expires. This option applies only if the `cof-feature` option is set to `true`.

cof-ci-wait-all

Default Value: `false`

Valid Values:

- `true` T-Server waits for responses from all T-Servers that might have the requested call data before updating the call data with the latest information.
- `false` T-Server updates the call data with the information received from the first positive response.

Changes Take Effect: Immediately

Specifies whether T-Server, after sending a request for matching call data, waits for responses from other T-Servers before updating the call data (such as `CallHistory`, `ConnID`, and `UserData`) for a potentially overflowed call. The waiting period is specified by the `cof-ci-req-tout` and `cof-rci-tout` options. This option applies only if the `cof-feature` option is set to `true`.

cof-feature

Default Value: `false`

Valid Values: `true`, `false`

Changes Take Effect: Immediately

Enables or disables the Inter Server Call Control/Call Overflow (ISCC/COF) feature.

cof-rci-tout

Default Value: `10 sec`

Valid Values: See “Timeout Value Format” on [page 206](#).

Changes Take Effect: For the next COF operation

Specifies the time interval that T-Server waits for call data from other T-Servers’ transactions. Counting starts when `cof-ci-req-tout` expires. This option applies only if the `cof-feature` option is set to `true`.

local-node-id

Default Value: `0`

Valid Values: `0` or any positive integer

Changes Take Effect: Immediately

This option, if enabled, checks all networked calls against the specified `NetworkNodeID` (the identity of the switch to which the call initially arrived). If the `NetworkNodeID` is the same as the value of this option, the request for call information is *not* sent. The default value of `0` disables the functionality of this

option. To establish an appropriate `NetworkNodeID`, specify a value other than the default. This option applies only if the `cof-feature` option is set to `true`.

Note: This option applies only to T-Server for Nortel Communication Server 2000/2100.

default-network-call-id-matching

Default Value: No default value

Valid Values: See the “T-Server-Specific Configuration Options” chapter for an option description for your T-Server

Changes Take Effect: Immediately

When a value for this option is specified, T-Server uses the `NetworkCallID` attribute for the ISCC/COF call matching.

To activate this feature, the `cof-feature` option must be set to `true`.

Note: SIP Server and several T-Servers support the `NetworkCallID` attribute for the ISCC/COF call matching in a way that requires setting this option to a specific value. For information about the option value that is specific for your T-Server, see the “T-Server-Specific Configuration Options” chapter of your *T-Server Deployment Guide*.

Event Propagation Options

compound-dn-representation

Default Value: `true`

Valid Values: `true`, `false`

Changes Take Effect: Immediately

Specifies which format T-Server uses to represent a DN when reporting an `OtherDN` or `ThirdPartyDN` attribute in event propagation messages.

When set to `true`, the `<switch>:DN` (compound) format is used. This option value supports backward compatibility for pre-8.x T-Server ISCC/EPP functionality and is provided for multi-site deployments where the same DNs are configured under several switches.

When set to `false`, the DN (non-compound) format is used. This option value ensures more transparent reporting of `OtherDN` or `ThirdPartyDN` attributes and is recommended for all single-site deployments, as well as for multi-site deployments that do not have the same DNs configured under several switches. This option applies only if the `event-propagation` option is set to `List`.

Note: Local DNs are always represented in the non-compound (DN) form.

epp-tout

Default Value: 0

Valid Values: See “Timeout Value Format” on [page 206](#).

Changes Take Effect: Immediately

Specifies the time interval during which T-Server attempts to resolve race conditions that may occur in deployments that use switch partitioning or intelligent trunks. This option applies only if the [event-propagation](#) option is set to `list`.

Note: If the time interval is not long enough to account for possible network switching delays, T-Server may produce duplicated events, such as events that are propagated by the ISCC and generated locally.

event-propagation

Default Value: `list`

Valid Values:

- `list` Changes in user data and party events are propagated to remote locations through call distribution topology.
- `off` The feature is disabled. Changes in user data and party events are not propagated to remote locations.

Changes Take Effect: Immediately

Specifies whether the Event Propagation feature is enabled.

Number Translation Option

inbound-translator-<n>

Default Value: No default value

Valid Value: Any valid name

Changes Take Effect: Immediately

Specifies the name of another configuration section as the value for the `inbound-translator` option. For example,

`inbound-translator-1 = ani-translator`

where `ani-translator` is the name of the configuration that describes the translation rules for inbound numbers.

GVP Integration Option

handle-vsp

Default Value: no

Valid Values:

requests	ISCC will process and adjust requests related to this DN and containing a Location attribute before submitting them to the service provider.
events	ISCC will process and adjust each event received from the service provider in response to a request containing a Location attribute before distributing the event to T-Server clients.
all	ISCC will process and adjust both events and requests.
no	No ISCC processing of such requests and events takes place.

Changes Take Effect: Immediately

Specifies if multi-site Call Data synchronization of virtual calls or simulated call flows is performed by T-Server or is left to an external application (Service Provider) that has registered for a DN with the AddressType attribute set to VSP (Virtual Service Provider).

backup-sync Section

The backup-synchronization section contains the configuration options that are used to support a high-availability (hot standby redundancy type) configuration.

This section must be called backup-sync.

Note: These options apply only to T-Servers that support the hot standby redundancy type.

addp-remote-timeout

Default Value: 0

Valid Values: Any integer from 0–3600

Changes Take Effect: Immediately

Specifies the time interval that the redundant T-Server waits for a response from this T-Server after sending a polling signal. The default value of 0 (zero) disables the functionality of this option. To establish an appropriate timeout, specify a value other than the default. This option applies only if the [protocol](#) option is set to addp.

addp-timeout

Default Value: 0

Valid Values: Any integer from 0–3600

Changes Take Effect: Immediately

Specifies the time interval that this T-Server waits for a response from another T-Server after sending a polling signal. The default value of 0 (zero) disables the functionality of this option. To establish an appropriate timeout, specify a value other than the default. This option applies only if the [protocol](#) option is set to `addp`.

addp-trace

Default Value: `off`

Valid Values:

`off`, `false`, `no` No trace (default).

`local`, `on`, `true`, `yes` Trace on this T-Server side only.

`remote` Trace on the redundant T-Server side only.

`full`, `both` Full trace (on both sides).

Changes Take Effect: Immediately

Specifies whether `addp` messages are traced in a log file, to what level the trace is performed, and in which direction. This option applies only if the [protocol](#) option is set to `addp`.

protocol

Default Value: `default`

Valid Values:

`default` The feature is not active.

`addp` Activates the Advanced Disconnect Detection Protocol.

Changes Take Effect: When the next connection is established

Specifies the name of the method used to detect connection failures. If you specify the `addp` value, you must also specify a value for the [addp-timeout](#), [addp-remote-timeout](#), and [addp-trace](#) options.

sync-reconnect-tout

Default Value: 20 sec

Valid Values: See “Timeout Value Format” on [page 206](#).

Changes Take Effect: Immediately

Specifies the time interval after which the backup T-Server attempts to reconnect to the primary server (for a synchronized link).

call-cleanup Section

The call-cleanup section contains the configuration options that are used to control detection and cleanup of stuck calls in T-Server. For more information on stuck call handling, refer to the “Stuck Call Management” chapter in the *Framework 8.1 Management Layer User’s Guide*.

This section must be called `call-cleanup`.

cleanup-idle-tout

Default Value: 0

Valid Values: See “Timeout Value Format” on [page 206](#).

Changes Take Effect: Immediately

Specifies the time interval that T-Server waits for a call to be updated from its last update. After this time elapses, if no new events about the call are received, T-Server clears this call as a stuck call, either by querying the switch (if a CTI link provides such capabilities) or by deleting the call information from memory unconditionally. The default value of 0 disables the stuck calls cleanup.

Note: If the call-cleanup functionality is enabled in T-Server for Avaya Communication Manager, the UCID (Universal Call ID) feature must be enabled on the switch as well. This allows the UCID to be generated and passed to T-Server.

notify-idle-tout

Default Value: 0

Valid Values: See “Timeout Value Format” on [page 206](#).

Changes Take Effect: Immediately

Specifies the time interval that T-Server waits for a call to be updated from its last update. After this time elapses, if no new events about the call are received, T-Server reports this call as a stuck call. The default value of 0 disables the stuck calls notification.

periodic-check-tout

Default Value: 10 min

Valid Values: See “Timeout Value Format” on [page 206](#).

Changes Take Effect: Immediately

Specifies the time interval for periodic checks for stuck calls. These checks affect both notification and cleanup functionality, and are made by checking the T-Server’s own call information with call information available in the switch. For performance reasons, T-Server does not verify whether the

`notify-idle-tout` or `cleanup-idle-tout` option has expired before performing this check.

Note: Setting this option to a value of less than a few seconds can affect T-Server performance.

Examples

This section presents examples of option settings in the `call-cleanup` section.

Example 1 `cleanup-idle-tout = 0`
`notify-idle-tout = 0`
`periodic-check-tout = 10`

With these settings, T-Server will not perform any checks for stuck calls.

Example 2 `cleanup-idle-tout = 0`
`notify-idle-tout = 5 min`
`periodic-check-tout = 10 min`

With these settings, T-Server performs checks every 10 minutes and sends notifications about all calls that have been idle for at least 5 minutes.

Example 3 `cleanup-idle-tout = 20 min`
`notify-idle-tout = 5 min`
`periodic-check-tout = 10 min`

With these settings, T-Server performs checks every 10 minutes, sends notifications about all calls that have been idle for at least 5 minutes, and attempts to clean up all calls that have been idle for more than 20 minutes.

Translation Rules Section

The section name is specified by the `inbound-translator-<n>` option. It contains options that define translation rules for inbound numbers.

You can choose any name for this section, provided that it matches the value of the section. Every option in this section corresponds to a rule and must conform to the format described below. You can configure as many rules as necessary to accommodate your business needs.

rule-<n>

Default Value: No default value

Valid Value: Any valid string in the following format:

`in-pattern=<input pattern value>;out-pattern=<output pattern value>`

Changes Take Effect: Immediately

Defines a rule to be applied to an inbound number. The two parts of the option value describe the input and output patterns in the rule. When configuring the

pattern values, follow the syntax defined in “Using ABNF for Rules” on [page 74](#). See “Configuring Number Translation” on [page 81](#) for examples of these rules as well as detailed instructions for creating rules for your installation. For example, a value for this configuration option might look like this:

```
rule-01 = in-pattern=0111#CABBB*ccD; out-pattern=ABD
```

security Section

The `security` section contains the configuration options that are used to configure secure data exchange between T-Servers and other Genesys components. Refer to the *Genesys 8.x Security Deployment Guide* for complete information on the security configuration.

Timeout Value Format

This section of the document describes the values to use for those T-Server common options that set various timeouts. The current format allows you to use fractional values and various time units for timeout settings.

For timeout-related options, you can specify any value that represents a time interval, provided that it is specified in one of the following formats:

```
[[hours:]minutes:]seconds][milliseconds]
```

or

```
[hours hr][minutes min][seconds sec][milliseconds msec]
```

Where a time unit name in *italic* (such as *hours*) is to be replaced by an integer value for this time unit.

Integer values with no measuring units are still supported, for compatibility with previous releases of T-Server. When you do not specify any measuring units, the units of the default value apply. For example, if the default value equals `60 sec`, specifying the value of `30` sets the option to 30 seconds.

Example 1

The following settings result in a value of 1 second, 250 milliseconds:

```
sync-reconnect-tout = 1.25
```

```
sync-reconnect-tout = 1 sec 250 msec
```

Example 2

The following settings result in a value of 1 minute, 30 seconds:

```
timeout = 1:30
```

```
timeout = 1 min 30 sec
```

Changes from Release 8.0 to 8.1

Table 14 lists the configuration options that:

- Are new or changed in the 8.1 release of T-Server
- Have been added or changed since the most recent 8.0 release of this document

If a configuration option has been replaced with another that enables the same functionality, the new option name and its location in this chapter are noted.

Table 14: Option Changes from Release 8.0 to 8.1

Option Name	Option Values	Type of Change	Details
TServer Section			
background-processing	true, false	See Details	Default value changed to true. See the option description on page 182 .

11

Common Configuration Options

Unless otherwise noted, the common configuration options that this chapter describes are common to all Genesys server applications and applicable to any Framework server component. This chapter includes the following sections:

- [Setting Configuration Options, page 209](#)
- [Mandatory Options, page 210](#)
- [log Section, page 210](#)
- [log-extended Section, page 224](#)
- [log-filter Section, page 226](#)
- [log-filter-data Section, page 226](#)
- [security Section, page 227](#)
- [sml Section, page 227](#)
- [common Section, page 229](#)
- [Changes from 8.0 to 8.1, page 229](#)

Note: Some server applications also support log options that are unique to them. For descriptions of a particular application's unique log options, refer to the chapter/document about that application.

Setting Configuration Options

Unless specified otherwise, set common configuration options in the Options of the Application object, using one of the following navigation paths:

- In Genesys Administrator—Application object > Options tab > Advanced View (Options)
- In Configuration Manager—Application object > Properties dialog box > Options tab

Warning! Configuration section names, configuration option names, and predefined option values are case-sensitive. Type them in Genesys Administrator or Configuration Manager exactly as they are documented in this chapter.

Mandatory Options

You do not have to configure any common options to start Server applications.

log Section

This section must be called `log`.

verbose

Default Value: `all`

Valid Values:

<code>all</code>	All log events (that is, log events of the Standard, Trace, Interaction, and Debug levels) are generated.
<code>debug</code>	The same as <code>all</code> .
<code>trace</code>	Log events of the Trace level and higher (that is, log events of the Standard, Interaction, and Trace levels) are generated, but log events of the Debug level are not generated.
<code>interaction</code>	Log events of the Interaction level and higher (that is, log events of the Standard and Interaction levels) are generated, but log events of the Trace and Debug levels are not generated.
<code>standard</code>	Log events of the Standard level are generated, but log events of the Interaction, Trace, and Debug levels are not generated.
<code>none</code>	No output is produced.

Changes Take Effect: Immediately

Determines whether a log output is created. If it is, specifies the minimum level of log events generated. The log events levels, starting with the highest priority level, are Standard, Interaction, Trace, and Debug. See also “Log Output Options” on [page 216](#).

Note: For definitions of the Standard, Interaction, Trace, and Debug log levels, refer to the *Framework 8.1 Management Layer User’s Guide*, *Framework 8.1 Genesys Administrator Help*, or to *Framework 8.1 Solution Control Interface Help*.

buffering

Default Value: `true`

Valid Values:

`true` Enables buffering.
`false` Disables buffering.

Changes Take Effect: Immediately

Turns on/off operating system file buffering. The option is applicable only to the `stderr` and `stdout` output (see [page 216](#)). Setting this option to `true` increases the output performance.

Note: When buffering is enabled, there might be a delay before log messages appear at the console.

segment

Default Value: `false`

Valid Values:

`false` No segmentation is allowed.
`<number> KB` or Sets the maximum segment size, in kilobytes. The minimum
`<number>` segment size is `100 KB`.
`<number> MB` Sets the maximum segment size, in megabytes.
`<number> hr` Sets the number of hours for the segment to stay open. The
 minimum number is 1 hour.

Changes Take Effect: Immediately

Specifies whether there is a segmentation limit for a log file. If there is, sets the mode of measurement, along with the maximum size. If the current log segment exceeds the size set by this option, the file is closed and a new one is created. This option is ignored if log output is not configured to be sent to a log file.

expire

Default Value: `false`

Valid Values:

`false` No expiration; all generated segments are stored.
`<number> file` or Sets the maximum number of log files to store. Specify a
`<number>` number from `1–1000`.
`<number> day` Sets the maximum number of days before log files are
 deleted. Specify a number from `1–100`.

Changes Take Effect: Immediately

Determines whether log files expire. If they do, sets the measurement for determining when they expire, along with the maximum number of files (segments) or days before the files are removed. This option is ignored if log output is not configured to be sent to a log file.

Note: If an option's value is set incorrectly—out of the range of valid values— it will be automatically reset to `10`.

keep-startup-fileDefault Value: `false`

Valid Values:

<code>false</code>	No startup segment of the log is kept.
<code>true</code>	A startup segment of the log is kept. The size of the segment equals the value of the <code>segment</code> option.
<code><number> KB</code>	Sets the maximum size, in kilobytes, for a startup segment of the log.
<code><number> MB</code>	Sets the maximum size, in megabytes, for a startup segment of the log.

Changes Take Effect: After restart

Specifies whether a startup segment of the log, containing the initial T-Server configuration, is to be kept. If it is, this option can be set to `true` or to a specific size. If set to `true`, the size of the initial segment will be equal to the size of the regular log segment defined by the `segment` option. The value of this option will be ignored if segmentation is turned off (that is, if the `segment` option set to `false`).

Note: This option applies only to T-Servers.

messagefile

Default Value: As specified by a particular application

Valid Values: `<string>.lms` (message file name)Changes Take Effect: Immediately, if an application cannot find its `*.lms` file at startup

Specifies the file name for application-specific log events. The name must be valid for the operating system on which the application is running. The option value can also contain the absolute path to the application-specific `*.lms` file. Otherwise, an application looks for the file in its working directory.

Warning! An application that does not find its `*.lms` file at startup cannot generate application-specific log events and send them to Message Server.

message_formatDefault Value: `short`

Valid Values:

<code>short</code>	An application uses compressed headers when writing log records in its log file.
<code>full</code>	An application uses complete headers when writing log records in its log file.

Changes Take Effect: Immediately

Specifies the format of log record headers that an application uses when writing logs in the log file. Using compressed log record headers improves application performance and reduces the log file's size.

With the value set to short:

- A header of the log file or the log file segment contains information about the application (such as the application name, application type, host type, and time zone), whereas single log records within the file or segment omit this information.
- A log message priority is abbreviated to Std, Int, Trc, or Dbg, for Standard, Interaction, Trace, or Debug messages, respectively.
- The message ID does not contain the prefix GCTI or the application type ID.

A log record in the full format looks like this:

```
2002-05-07T18:11:38.196 Standard localhost cfg_dbserver GCTI-00-05060
Application started
```

A log record in the short format looks like this:

```
2002-05-07T18:15:33.952 Std 05060 Application started
```

Note: Whether the full or short format is used, time is printed in the format specified by the `time_format` option.

time_convert

Default Value: Local

Valid Values:

local	The time of log record generation is expressed as a local time, based on the time zone and any seasonal adjustments. Time zone information of the application's host computer is used.
utc	The time of log record generation is expressed as Coordinated Universal Time (UTC).

Changes Take Effect: Immediately

Specifies the system in which an application calculates the log record time when generating a log file. The time is converted from the time in seconds since the Epoch (00:00:00 UTC, January 1, 1970).

time_format

Default Value: time

Valid Values:

time	The time string is formatted according to the HH:MM:SS.sss (hours, minutes, seconds, and milliseconds) format.
locale	The time string is formatted according to the system's locale.
ISO8601	The date in the time string is formatted according to the ISO 8601 format. Fractional seconds are given in milliseconds.

Changes Take Effect: Immediately

Specifies how to represent, in a log file, the time when an application generates log records.

A log record's time field in the ISO 8601 format looks like this:

2001-07-24T04:58:10.123

print-attributes

Default Value: `false`

Valid Values:

`true` Attaches extended attributes, if any exist, to a log event sent to log output.

`false` Does not attach extended attributes to a log event sent to log output.

Changes Take Effect: Immediately

Specifies whether the application attaches extended attributes, if any exist, to a log event that it sends to log output. Typically, log events of the Interaction log level and Audit-related log events contain extended attributes. Setting this option to `true` enables audit capabilities, but negatively affects performance. Genesys recommends enabling this option for Solution Control Server and Configuration Server when using audit tracking. For other applications, refer to *Genesys 8.1 Combined Log Events Help* to find out whether an application generates Interaction-level and Audit-related log events; if it does, enable the option only when testing new interaction scenarios.

check-point

Default Value: 1

Valid Values: 0–24

Changes Take Effect: Immediately

Specifies, in hours, how often the application generates a check point log event, to divide the log into sections of equal time. By default, the application generates this log event every hour. Setting the option to 0 prevents the generation of check-point events.

memory

Default Value: No default value

Valid Values: `<string>` (memory file name)

Changes Take Effect: Immediately

Specifies the name of the file to which the application regularly prints a snapshot of the memory output, if it is configured to do this (see “Log Output Options” on [page 216](#)). The new snapshot overwrites the previously written data. If the application terminates abnormally, this file will contain the latest

log messages. Memory output is not recommended for processors with a CPU frequency lower than 600 MHz.

Note: If the file specified as the memory file is located on a network drive, an application does not create a snapshot file (with the extension `*.memory.log`).

memory-storage-size

Default Value: 2 MB

Valid Values:

<number> KB or <number> The size of the memory output, in kilobytes.
The minimum value is 128 KB.

<number> MB The size of the memory output, in megabytes.
The maximum value is 64 MB.

Changes Take Effect: When memory output is created

Specifies the buffer size for log output to the memory, if configured. See also “Log Output Options” on [page 216](#).

spool

Default Value: The application’s working directory

Valid Values: <path> (the folder, with the full path to it)

Changes Take Effect: Immediately

Specifies the folder, including full path to it, in which an application creates temporary files related to network log output. If you change the option value while the application is running, the change does not affect the currently open network output.

compatible-output-priority

Default Value: false

Valid Values:

true The log of the level specified by “Log Output Options” is sent to the specified output.

false The log of the level specified by “Log Output Options” and higher levels is sent to the specified output.

Changes Take Effect: Immediately

Specifies whether the application uses 6.x output logic. For example, you configure the following options in the log section for a 6.x application and for a 7.x application:

```
[log]
```

```
verbose = all
```

```
debug = file1
```

```
standard = file2
```

The log file content of a 6.x application is as follows:

- `file1` contains Debug messages only.
- `file2` contains Standard messages only.

The log file content of a 7.x application is as follows:

- `file1` contains Debug, Trace, Interaction, and Standard messages.
- `file2` contains Standard messages only.

If you set `compatible-output-priority` to `true` in the 7.x application, its log file content will be the same as for the 6.x application.

Warning! Genesys does not recommend changing the default value of this option unless you have specific reasons to use the 6.x log output logic—that is, to mimic the output priority as implemented in releases 6.x. Setting this option to `true` affects log consistency.

Log Output Options

To configure log outputs, set log level options (`all`, `alarm`, `standard`, `interaction`, `trace`, and/or `debug`) to the desired types of log output (`stdout`, `stderr`, `network`, `memory`, and/or `[filename]`, for log file output).

You can use:

- One log level option to specify different log outputs.
- One log output type for different log levels.
- Several log output types simultaneously, to log events of the same or different log levels.

You must separate the log output types by a comma when you are configuring more than one output for the same log level. See “Examples” on [page 220](#).

Warnings!

- If you direct log output to a file on the network drive, an application does not create a snapshot log file (with the extension `*.snapshot.log`) in case it terminates abnormally.
- Directing log output to the console (by using the `stdout` or `stderr` settings) can affect application performance. Avoid using these log output settings in a production environment.

Note: The log output options are activated according to the setting of the `verbose` configuration option.

all

Default Value: No default value

Valid Values (log output types):

<code>stdout</code>	Log events are sent to the Standard output (<code>stdout</code>).
<code>stderr</code>	Log events are sent to the Standard error output (<code>stderr</code>).
<code>network</code>	Log events are sent to Message Server, which can reside anywhere on the network. Message Server stores the log events in the Log Database. Setting the <code>all</code> log level option to the <code>network</code> output enables an application to send log events of the <code>Standard</code> , <code>Interaction</code> , and <code>Trace</code> levels to Message Server. Debug-level log events are neither sent to Message Server nor stored in the Log Database.
<code>memory</code>	Log events are sent to the memory output on the local disk. This is the safest output in terms of the application performance.
<code>[filename]</code>	Log events are stored in a file with the specified name. If a path is not specified, the file is created in the application's working directory.

Changes Take Effect: Immediately

Specifies the outputs to which an application sends all log events. The log output types must be separated by a comma when more than one output is configured. For example:

```
all = stdout, logfile
```

Note: To ease the troubleshooting process, consider using unique names for log files that different applications generate.

alarm

Default Value: No default value

Valid Values (log output types):

<code>stdout</code>	Log events are sent to the Standard output (<code>stdout</code>).
<code>stderr</code>	Log events are sent to the Standard error output (<code>stderr</code>).
<code>network</code>	Log events are sent to Message Server, which resides anywhere on the network, and Message Server stores the log events in the Log Database.
<code>memory</code>	Log events are sent to the memory output on the local disk. This is the safest output in terms of the application performance.
<code>[filename]</code>	Log events are stored in a file with the specified name. If a path is not specified, the file is created in the application's working directory.

Changes Take Effect: Immediately

Specifies the outputs to which an application sends the log events of the Alarm level. The log output types must be separated by a comma when more than one output is configured. For example:

```
standard = stderr, network
```

standard

Default Value: No default value

Valid Values (log output types):

<code>stdout</code>	Log events are sent to the Standard output (<code>stdout</code>).
<code>stderr</code>	Log events are sent to the Standard error output (<code>stderr</code>).
<code>network</code>	Log events are sent to Message Server, which can reside anywhere on the network. Message Server stores the log events in the Log Database.
<code>memory</code>	Log events are sent to the memory output on the local disk. This is the safest output in terms of the application performance.
<code>[filename]</code>	Log events are stored in a file with the specified name. If a path is not specified, the file is created in the application's working directory.

Changes Take Effect: Immediately

Specifies the outputs to which an application sends the log events of the Standard level. The log output types must be separated by a comma when more than one output is configured. For example:

```
standard = stderr, network
```

interaction

Default Value: No default value

Valid Values (log output types):

<code>stdout</code>	Log events are sent to the Standard output (<code>stdout</code>).
<code>stderr</code>	Log events are sent to the Standard error output (<code>stderr</code>).
<code>network</code>	Log events are sent to Message Server, which can reside anywhere on the network. Message Server stores the log events in the Log Database.
<code>memory</code>	Log events are sent to the memory output on the local disk. This is the safest output in terms of the application performance.
<code>[filename]</code>	Log events are stored in a file with the specified name. If a path is not specified, the file is created in the application's working directory.

Changes Take Effect: Immediately

Specifies the outputs to which an application sends the log events of the Interaction level and higher (that is, log events of the Standard and Interaction levels). The log outputs must be separated by a comma when more than one output is configured. For example:

```
interaction = stderr, network
```

trace

Default Value: No default value

Valid Values (log output types):

<code>stdout</code>	Log events are sent to the Standard output (<code>stdout</code>).
<code>stderr</code>	Log events are sent to the Standard error output (<code>stderr</code>).
<code>network</code>	Log events are sent to Message Server, which can reside anywhere on the network. Message Server stores the log events in the Log Database.
<code>memory</code>	Log events are sent to the memory output on the local disk. This is the safest output in terms of the application performance.
<code>[filename]</code>	Log events are stored in a file with the specified name. If a path is not specified, the file is created in the application's working directory.

Changes Take Effect: Immediately

Specifies the outputs to which an application sends the log events of the Trace level and higher (that is, log events of the Standard, Interaction, and Trace levels). The log outputs must be separated by a comma when more than one output is configured. For example:

```
trace = stderr, network
```

debug

Default Value: No default value

Valid Values (log output types):

<code>stdout</code>	Log events are sent to the Standard output (<code>stdout</code>).
<code>stderr</code>	Log events are sent to the Standard error output (<code>stderr</code>).
<code>memory</code>	Log events are sent to the memory output on the local disk. This is the safest output in terms of the application performance.
<code>[filename]</code>	Log events are stored in a file with the specified name. If a path is not specified, the file is created in the application's working directory.

Changes Take Effect: Immediately

Specifies the outputs to which an application sends the log events of the Debug level and higher (that is, log events of the Standard, Interaction, Trace, and Debug levels). The log output types must be separated by a comma when more than one output is configured—for example:

```
debug = stderr, /usr/local/genesys/logfile
```

Note: Debug-level log events are never sent to Message Server or stored in the Log Database.

Log File Extensions

You can use the following file extensions to identify log files that an application creates for various types of output:

- *.log—Assigned to log files when you configure output to a log file. For example, if you set `standard = confservlog` for Configuration Server, it prints log messages into a text file called `confservlog.<time_stamp>.log`.
- *.qsp—Assigned to temporary (spool) files when you configure output to the network but the network is temporarily unavailable. For example, if you set `standard = network` for Configuration Server, it prints log messages into a file called `confserv.<time_stamp>.qsp` during the time the network is not available.
- *.snapshot.log—Assigned to files that contain the output snapshot when you configure output to a log file. The file contains the last log messages that an application generates before it terminates abnormally. For example, if you set `standard = confservlog` for Configuration Server, it prints the last log message into a file called `confserv.<time_stamp>.snapshot.log` in case of failure.

Note: Provide *.snapshot.log files to Genesys Technical Support when reporting a problem.

- *.memory.log—Assigned to log files that contain the memory output snapshot when you configure output to memory and redirect the most recent memory output to a file. For example, if you set `standard = memory` and `memory = confserv` for Configuration Server, it prints the latest memory output to a file called `confserv.<time_stamp>.memory.log`.

Examples

This section presents examples of a log section that you might configure for an application when that application is operating in production mode and in two lab modes, debugging and troubleshooting.

Production Mode Log Section

```
[log]
verbose = standard
standard = network, logfile
```

With this configuration, an application only generates the log events of the Standard level and sends them to Message Server, and to a file named `logfile`, which the application creates in its working directory. Genesys recommends that you use this or a similar configuration in a production environment.

Warning! Directing log output to the console (by using the `stdout` or `stderr` settings) can affect application performance. Avoid using these log output settings in a production environment.

Lab Mode Log Section

```
[log]
verbose = all
all = stdout, /usr/local/genesys/logfile
trace = network
```

With this configuration, an application generates log events of the Standard, Interaction, Trace, and Debug levels, and sends them to the standard output and to a file named `logfile`, which the application creates in the `/usr/local/genesys/` directory. In addition, the application sends log events of the Standard, Interaction, and Trace levels to Message Server. Use this configuration to test new interaction scenarios in a lab environment.

Failure-Troubleshooting Log Section

```
[log]
verbose = all
standard = network
all = memory
memory = logfile
memory-storage-size = 32 MB
```

With this configuration, an application generates log events of the Standard level and sends them to Message Server. It also generates log events of the Standard, Interaction, Trace, and Debug levels, and sends them to the memory output. The most current log is stored to a file named `logfile`, which the application creates in its working directory. Increased memory storage allows an application to save more of the log information generated before a failure.

Note: If you are running an application on UNIX, and you do not specify any files in which to store the memory output snapshot, a core file that the application produces before terminating contains the most current application log. Provide the application's core file to Genesys Technical Support when reporting a problem.

Debug Log Options

The options in this section enable you to generate Debug logs containing information about specific operations of an application.

x-conn-debug-open

Default Value: 0

Valid Values:

- 0 Log records are not generated.
- 1 Log records are generated.

Changes Take Effect: After restart

Generates Debug log records about “open connection” operations of the application.

Warning! Use this option only when requested by Genesys Technical Support.

x-conn-debug-select

Default Value: 0

Valid Values:

- 0 Log records are not generated.
- 1 Log records are generated.

Changes Take Effect: After restart

Generates Debug log records about “socket select” operations of the application.

Warning! Use this option only when requested by Genesys Technical Support.

x-conn-debug-timers

Default Value: 0

Valid Values:

- 0 Log records are not generated.
- 1 Log records are generated.

Changes Take Effect: After restart

Generates Debug log records about the timer creation and deletion operations of the application.

Warning! Use this option only when requested by Genesys Technical Support.

x-conn-debug-write

Default Value: 0

Valid Values:

- 0 Log records are not generated.
- 1 Log records are generated.

Changes Take Effect: After restart

Generates Debug log records about “write” operations of the application.

Warning! Use this option only when requested by Genesys Technical Support.

x-conn-debug-security

Default Value: 0

Valid Values:

0 Log records are not generated.

1 Log records are generated.

Changes Take Effect: After restart

Generates Debug log records about security-related operations, such as Transport Layer Security and security certificates.

Warning! Use this option only when requested by Genesys Technical Support.

x-conn-debug-api

Default Value: 0

Valid Values:

0 Log records are not generated.

1 Log records are generated.

Changes Take Effect: After restart

Generates Debug log records about connection library function calls.

Warning! Use this option only when requested by Genesys Technical Support.

x-conn-debug-dns

Default Value: 0

Valid Values:

0 Log records are not generated.

1 Log records are generated.

Changes Take Effect: After restart

Generates Debug log records about DNS operations.

Warning! Use this option only when requested by Genesys Technical Support.

x-conn-debug-all

Default Value: 0

Valid Values:

- 0 Log records are not generated.
- 1 Log records are generated.

Changes Take Effect: After restart

Generates Debug log records about open connection, socket select, timer creation and deletion, write, security-related, and DNS operations, and connection library function calls. This option is the same as enabling or disabling all of the previous x-conn-debug-`<op type>` options.

Warning! Use this option only when requested by Genesys Technical Support.

log-extended Section

This section must be called log-extended.

level-reassign-`<eventID>`Default Value: Default value of log event `<eventID>`

Valid Values:

- alarm The log level of log event `<eventID>` is set to Alarm.
- standard The log level of log event `<eventID>` is set to Standard.
- interaction The log level of log event `<eventID>` is set to Interaction.
- trace The log level of log event `<eventID>` is set to Trace.
- debug The log level of log event `<eventID>` is set to Debug.
- none Log event `<eventID>` is not recorded in a log.

Changes Take Effect: Immediately

Specifies a log level for log event `<eventID>` that is different than its default level, or disables log event `<eventID>` completely. If no value is specified, the log event retains its default level. This option is useful when you want to customize the log level for selected log events.

These options can be deactivated with the option [level-reassign-disable](#).

Warning! Use caution when making these changes in a production environment.

Depending on the log configuration, changing the log level to a higher priority may cause the log event to be logged more often or to a greater number of outputs. This could affect system performance.

Likewise, changing the log level to a lower priority may cause the log event to be not logged at all, or to be not logged to specific outputs, thereby losing important information. The same applies to any alarms associated with that log event.

In addition to the preceding warning, take note of the following:

- Logs can be customized only by release 7.6 or later applications.
- When the log level of a log event is changed to any level except none, it is subject to the other settings in the [log] section at its new level. If set to none, it is not logged and is therefore not subject to any log configuration.
- Using this feature to change the log level of a log changes only its priority; it does not change how that log is treated by the system. For example, increasing the priority of a log to Alarm level does not mean that an alarm will be associated with it.
- Each application in a High Availability (HA) pair can define its own unique set of log customizations, but the two sets are not synchronized with each other. This can result in different log behavior depending on which application is currently in primary mode.
- This feature is not the same as a similar feature in Universal Routing Server (URS) release 7.2 or later. In this Framework feature, the priority of log events are customized. In the URS feature, the priority of debug messages only are customized. Refer to the *Universal Routing Reference Manual* for more information about the URS feature.
- You cannot customize any log event that is not in the unified log record format. Log events of the Alarm, Standard, Interaction, and Trace levels feature the same unified log record format.

Example

This is an example of using customized log level settings, subject to the following log configuration:

```
[log]
verbose=interaction
all=stderr
interaction=log_file
standard=network
```

Before the log levels of the log are changed:

- Log event 1020, with default level standard, is output to `stderr` and `log_file`, and sent to Message Server.
- Log event 2020, with default level standard, is output to `stderr` and `log_file`, and sent to Message Server.
- Log event 3020, with default level trace, is output to `stderr`.
- Log event 4020, with default level debug, is output to `stderr`.

Extended log configuration section:

```
[log-extended]
level-reassign-1020=none
level-reassign-2020=interaction
level-reassign-3020=interaction
level-reassign-4020=standard
```

After the log levels are changed:

- Log event 1020 is disabled and not logged.
- Log event 2020 is output to `stderr` and `log_file`.
- Log event 3020 is output to `stderr` and `log_file`.
- Log event 4020 is output to `stderr` and `log_file`, and sent to Message Server.

level-reassign-disable

Default Value: `false`

Valid Values: `true`, `false`

Changes Take Effect: Immediately

When this option is set to `true`, the original (default) log level of all log events in the `[log-extended]` section are restored. This option is useful when you want to use the default levels, but not delete the customization statements.

log-filter Section

The `log-filter` section contains configuration options used to define the default treatment of filtering data in log output. This section contains one configuration option, `default-filter-type`. Refer to the chapter “Hide Selected Data in Logs” in the *Genesys 8.x Security Deployment Guide* for complete information about this option.

log-filter-data Section

The `log-filter-data` section contains configuration options used to define the treatment of filtering data in log output on a key-by-key basis. This section contains one configuration option in the form of `<key name>`. Refer to the

chapter “Hide Selected Data in Logs” in the *Genesys 8.1 Security Deployment Guide* for complete information about this option.

security Section

The `security` section contains configuration options used to specify security elements for your system. In addition to other options that may be required by your application, this section contains the configuration option `disable-rbac`, which is used to enable or disable Role-Based Access Control for an application. Refer to the chapter “Role-Based Access Control” in the *Genesys 8.x Security Deployment Guide* for complete information about this option.

sml Section

This section must be called `sml`.

Options in this section are defined in the Annex of the `Application` object, as follows:

- in Genesys Administrator—`Application` object > `Options` tab > `Advanced View` (Annex)
- in Configuration Manager—`Application` object > `Properties` dialog box > `Annex` tab

Warning! Use the first three options in this section (`heartbeat-period`, `heartbeat-period-thread-class-<n>`, and `hangup-restart`) with great care, and only with those applications of which support for this functionality has been announced. Failure to use these options properly could result in unexpected behavior, from ignoring the options to an unexpected restart of the application.

heartbeat-period

Default Value: None

Valid Values:

- | | |
|-----------------------|---|
| <code>0</code> | This method of detecting an unresponsive application is not used by this application. |
| <code>3-604800</code> | Length of timeout, in seconds; equivalent to 3 seconds–7 days. |

Changes Take Effect: Immediately

Specifies the maximum amount of time, in seconds, in which heartbeat messages are expected from an application. If Local Control Agent (LCA) does not receive a heartbeat message from the application within this period, it assumes the application is not responding and carries out corrective action.

This option can also be used to specify the maximum heartbeat interval for threads registered with class zero (0). This thread class is reserved for use by the Management Layer only.

If this option is not configured or is set to zero (0), heartbeat detection is not used by this application.

heartbeat-period-thread-class-<n>

Default Value: None

Valid Values:

- 0 Value specified by `heartbeat-period` in application is used.
- 3-604800 Length of timeout, in seconds; equivalent to 3 seconds–7 days.

Changes Take Effect: Immediately

Specifies the maximum amount of time, in seconds, in which heartbeat messages are expected from a thread of class <n> registered by an application. If a heartbeat message from the thread is not received within this period, the thread is assumed to be not responding, and therefore, the application is unable to provide service.

If this option is not configured or is set to zero (0), but the application has registered one or more threads of class <n>, the value specified by the value of `heartbeat-period` for the application will also be applied to these threads.

There are no specific thread classes within IVR Server.

hangup-restart

Default Value: true

Valid Values: true, false

Changes Take Effect: Immediately

If set to true (the default), specifies that LCA is to restart the unresponsive application immediately, without any further interaction from Solution Control Server.

If set to false, specifies that LCA is only to generate a notification that the application has stopped responding.

suspending-wait-timeout

Default Value: 10

Valid Values: 5-600

Changes Take Effect: Immediately

Specifies a timeout (in seconds) after the Stop Graceful command is issued to an application during which the status of the application should change to Suspending if the application supports graceful shutdown. If the status of the application does not change to Suspending before the timeout expires, it is assumed that the application does not support graceful shutdown, and it is stopped ungracefully.

Use this option if you are unsure whether the Application supports graceful shutdown.

Note: Genesys recommends that you do not set this option for any Management Layer component (Configuration Server, Message Server, Solution Control Server, or SNMP Master Agent) or any DB Server. These components by definition do not support graceful shutdown, so this option is not required.

common Section

This section must be called `common`.

enable-async-dns

Default Value: `off`

Valid Values:

`off` Disables asynchronous processing of DNS requests.
`on` Enables asynchronous processing of DNS requests.

Changes Take Effect: Immediately

Enables the asynchronous processing of DNS requests such as, for example, host-name resolution.

Warnings! • Use this option only when requested by Genesys Technical Support.
• Use this option only with T-Servers.

rebind-delay

Default Value: `10`

Valid Values: `0–600`

Changes Take Effect: After restart

Specifies the delay, in seconds, between socket-bind operations that are being executed by the server. Use this option if the server has not been able to successfully occupy a configured port.

Warning! Use this option only when requested by Genesys Technical Support.

Changes from 8.0 to 8.1

There are no changes in common configuration options between 8.0 and 8.1 releases.

12

IVR Configuration Options

This chapter describes the configuration options that are unique to the IVR Server for IVR Interface Option 8.1. It contains the following sections:

- [TServer_IVR Options, page 231](#)
- [I-Server Options, page 244](#)
- [IVR_Driver Options, page 248](#)
- [IVR Annex Options, page 255](#)
- [IVR Port Annex Options, page 259](#)
- [Changes from Release 8.0 to 8.1, page 260](#)

TServer_IVR Options

TServer_IVR configuration options are defined in the Options of the TServer_IVR Application object, using one of the following navigation paths:

- In Genesys Administrator—Application object > Options tab > Advanced View (Options)
- In Configuration Manager—Application object > Properties dialog box > Options tab

For ease of reference, the options have been arranged in alphabetical order within their corresponding sections:

- “CallIdSap” on [page 233](#)
- “gli” on [page 233](#)
- “gli_server” on [page 234](#)
- “gli_server_group_<n>” on [page 234](#)
- “IServer” on [page 236](#)
- “IServerGLMSap” on [page 238](#)
- “pgf” on [page 239](#)

- “pgf-debug” on [page 239](#)
- “Timers” on [page 239](#)
- “TServerClientSap” on [page 242](#)
- “XmlSap” on [page 243](#)

Common Options

Some of the options that can be configured for the TServer_IVR application are *common options*, which means that they are shared with other types of T-Server applications. For the release 8.1 versions of this *System Administrator's Guide*, these common options have been omitted from this chapter.

[Table 15](#) provides information about the options that have been moved (grouped according to section), including links to their new locations and information about them that is specific to IVR Server.

Table 15: Common Options Information

Section Name	IVR Server Configuration Notes
extrouter	You should configure external routing on the TServer_IVR application only if you are using the IVR-In-Front or <i>dual</i> (IVR-In-Front and IVR-Behind-Switch) configuration mode. If you are using <i>only</i> the IVR-Behind-Switch configuration mode, you must configure external routing on the premise T-Server application. See “extrouter Section” on page 191 .
license	You should specify the num-sdn-licenses option for the TServer_IVR application only in the following circumstances: <ul style="list-style-type: none"> • The DNs that IVR Server uses are configured as Extensions and not as Voice Treatment Ports. • IVR Server is configured in IVR Network T-Server mode and uses the XML GenSpec protocols. See “license Section” on page 187 .
Log	See Chapter 11, “Common Configuration Options,” on page 209 .
TServer	See “TServer Section” on page 182 .

CallIdSap

This section must be called `CallIdSap`.

input-network-call-id-key

Default Value: None

Valid Value: `XML.Message.GctiMsg.CallId.Val`

Changes Take Effect: After `TServer_IVR` is restarted

Specifies the property map element where the call ID for inbound messages is stored.

gli

This section must be called `gli`.

gli-keep-alive-interval

Default Value: -1

Valid Values: Any positive integer

Changes Take Effect: Immediately

Specifies the amount of time (in seconds) before a `KeepAliveRequest` is sent. A value of -1 disables the sending of `KeepAliveRequests`.

gli-keep-alive-tries

Default Value: 1

Valid Values: Any positive integer

Changes Take Effect: Immediately

Specifies the number of times that the interval specified by `gli-keep-alive-interval` can lapse without receiving either a message or a `KeepAliveResponse`. When this value is reached, the link is closed.

gli-mode

Default Value: None

Valid Values: `server`, `client`

Changes Take Effect: After `TServer_IVR` is restarted

Specifies whether the GDI Link Interface (GLI) layer acts as a client or a server. If set to `server`, the IVR Server uses the options in the `gli_server` section (see [page 234](#)). Because the IVR Server always acts in the server role, this option must be set to `server`, and the `gli_server` section must also be provided.

gli-reconnect-delay

Default Value: 5

Valid Values: Any positive integer

Changes Take Effect: Immediately

Specifies the amount of time (in seconds) before a reconnection is attempted.

This option affects both server and client connections.

gli_server

This section must be called `gli_server`.

gli-<n>-servers

Default Value: None

Valid Values: 1–8

Changes Take Effect: Immediately

Used in `circuit` mode only. Specifies the number of server group sections to read. These server group sections are named `gli_server_group_1`, `gli_server_group_2`, and so on. Each server group section contains a `gli-server-address` and a `gli-client-list` option.

gli-server-mode

Default Value: None

Valid Value: `circuit`

Changes Take Effect: Immediately

When set to `circuit`, the `gli-<n>-servers` option is used.

gli_server_group_<n>

This section must be called `gli_server_group_<n>`.

gli-circuit-failover

Default Value: `on`

Valid Values: `on`, `off`

Changes Take Effect: After T-Server is restarted

Can be set to `on` only if the T-Server clients (SCPs) share call context; otherwise, it must be set to `off`.

gli-client-list

Default Value: None

Valid Values: Comma separated list of addresses in the form `host:port`

Changes Take Effect: Immediately

Specifies the clients that are allowed to connect to the server. The value is in a comma-separated format, where each client consists of the host name and port

number. If no value is present, any client may connect. Otherwise, an exact match (through DNS or BIND) is required. Using an asterisk matches any port.

gli-server-address

Default Value: None

Valid Values: Comma separated list of addresses in the form host:port

Changes Take Effect: After TServer_IVR is restarted

Specifies the address to use when creating the server's listen socket. You can specify more than one address by entering a comma-separated list of addresses, where each address is composed of the host name or IP address, and the TCP/IP port number, in the following format:

<host name or IP_address>:<TCP/IP port #>

gli-tls-cert

Default Value: None

Valid Values: N/A

Changes Take Effect: After TServer_IVR is restarted

For Windows, contains the thumbprint that is obtained from a user certificate generated for the host. For UNIX, contains the path and file name to a .pem encoded file that contains the host certificate.

gli-tls-cert-key

Default Value: None

Valid Values: N/A

Changes Take Effect: After TServer_IVR is restarted

For UNIX only, contains the path and file name to a .pem encoded file that contains the host private key.

gli-tls-trusted-ca

Default Value: None

Valid Values: N/A

Changes Take Effect: After TServer_IVR is restarted

For UNIX only, contains the path and file name to a .pem encoded file that contains the Certificate Authority (CA) certificate.

IServer

This section must be called IServer.

active-release

Default Value: true

Valid Values: true, false

Changes Take Effect: Immediately

Sets the default `GCTIActiveRelease` value when it is not present in an `EndCall` message. The `GCTIActiveRelease` flag determines whether IVR Server sends a `TReleaseCall` T-Library request to the T-Server in order to actively terminate the call. If this option is set to true, then a `TReleaseCall` request is sent. If it is set to false, then a `TReleaseCall` request is not sent.

app-name

Default Value: None

Valid Values: Any valid I-Server application object name

Changes Take Effect: After TServer_IVR is restarted

Specifies the name of the I-Server application object in the Configuration Layer. This option is required for IVR-In-Front and IVR-Behind-Switch configurations, but is not used for IVR Network T-Server configurations.

called-num-subset

Default Value: 0

Valid Values: Any positive integer

Changes Take Effect: After TServer_IVR is restarted

For an IVR Network T-Server configuration, specifies the number of rightmost digits of `callNum` that are used as `AttributeThisDN` in T-Library messages. This option is provided for the purpose of compatibility with Network T-Server for XML-based GenSpec.

dtd-file

Default Value: `IServer.dtd`

Valid Values: Any relative or absolute path to a .dtd file

Changes Take Effect: After TServer_IVR is restarted

Specifies the name of the .dtd file to be included in XML messages that IVR Server sends.

flow-control

Default Value: off

Valid Values: on, off

Changes Take Effect: Immediately

Enables flow control for a related IVR Server. If set to on, no new calls can be started at that IVR Server; however, existing calls proceed normally.

peer-list

Default Value: No

Valid Values: No, or a comma-delimited list of peer

TServer_IVR_application_names

Changes Take Effect: After the IVR Driver is restarted.

Specifies the names of all TServer_IVRs that are part of one load-balance group. If there are at least two names, then operation will be in Peer mode.

A value must be configured on each of the TServer_IVRs.

peer-mode-dn

Default Value: PeerModeDN

Valid Values: A DN configured on the virtual switch that is the one configured for the TServer_IVR.

Changes Take Effect: After the IVR Driver is restarted.

Specifies the DN (configured as an extension) that will be used only for the distribution of peer poll messages.

A value must be configured on each of the TServer_IVRs.

peer-mode-timer

Default Value: 20

Valid Values: Any integer from 5 upwards

Changes Take Effect: After the IVR Driver is restarted.

Specifies the time (in seconds) between poll messages being sent.

A value must be configured on each of the TServer_IVRs.

report-dn-status

Default Value: false

Valid Values: true, false

Changes Take Effect: At startup

Specifies whether EventDNOutOfService/EventDNBackInService messages are generated.

route-in-place

Default Value: false

Valid Values: true, false

Changes Take Effect: Immediately

Enables IVR Server to support the simulated re-routing between Routing Points configured for IVR Server's virtual switch or virtual Routing Points configured on the premise switch for the IVR Server in Behind mode.

If the value is set to true for the Routing Points or virtual Routing Points, then for every EventRouteRequest which returns a RequestRouteCall with the destination of another Routing Point on the virtual switch or virtual Routing Point on the premise switch, IVR Server issues EventRouteUsed for the new

Routing Point or virtual Routing Point, and an EventRouteRequest on the new Routing Point or virtual Routing Point. This sequence is continued until a RequestRouteCall to a destination other than a Routing Point on the virtual switch or virtual Routing Point on the premise switch is returned.

transport-port

Default Value: No default value

Valid Values: Any valid TCP/IP port

Changes Take Effect: After TServer_IVR is restarted

Specifies the port number of the client-side port that the TServer_IVR application uses to connect to Configuration Server.

IServerGLMSap

This section must be called IServerGLMSap.

checkout-interval

Default Value: 600

Valid Values: Any value from 600 to 3600

Changes Take Effect: Immediately

Specifies the interval (in seconds) at which the IVR Server attempts to request license updates from the FlexLM license server for the maximum call usage during that interval. IVR Server keeps track of the maximum number of concurrent calls during each interval, and at the end of the interval, it sends a message to FlexLM to update the licenses used by the IVR Server to the maximum concurrent value for that interval.

The minimum possible value is 600 seconds, and the maximum possible value is 3600 seconds. If you set a value that is outside this range, it is ignored, and the default is used instead.

operation-mode

Default Value: None

Valid Values: NTS, IVR

Changes Take Effect: After TServer_IVR is restarted

Specifies the IVR Server deployment mode. IVR Server cannot be started until you set this option.

Set the value to NTS for an IVR Network T-Server configuration. This enables the IVR Server to use the GenSpec XML protocols.

Set the value to IVR for an IVR-In-Front or IVR-Behind-Switch configuration.

pgf

This section must be called `pgf`.

ptc-file

Default Value: `tserver.ptc`

Valid Values: Any valid relative or absolute path to the `I-Server.smx` file

Changes Take Effect: After `TServer_IVR` is restarted

Specifies the file name for the `.ptc` or `.smx` file for this `TServer_IVR`.

pgf-debug

This section must be called `pgf-debug`.

If the section is *not* present, default debugging level(s) will be enabled. If the section is present, the `debug` option will determine the debug levels.

debug

Default Value: `default:All`

Valid Values: Any valid log level

Changes Take Effect: Immediately

Specifies the log level output.

hide-user-data

Default Value: `false`

Valid Values: `true`, `false`

Changes Take Effect: After `TServer_IVR` is restarted

Specifies whether user data hiding is enabled. When set to `true`, the user data is not displayed in the logs. When set to `false` (the default), the user data is displayed in the logs.

Timers

This section must be called `Timers`.

Valid Units of Time

The default unit of time for all timers is milliseconds (`ms`). This means that if you do not supply a unit of time when you set an option value, `ms` is assumed.

The valid units of time are as follows:

- `ms` = milliseconds
- `s` = seconds
- `h` = hours
- `d` = days

Call Timeout

Default Value: 8 h

Valid Values: Any positive integer with a valid unit of time (see [“Valid Units of Time”](#))

Changes Take Effect: After TServer_IVR is restarted

Specifies the amount of time that a call can remain active before it is ended.

CME Update Timeout

Default Value: 2 s

Valid Values: Any positive integer with a valid unit of time (see [“Valid Units of Time”](#))

Changes Take Effect: After TServer_IVR is restarted

Specifies the length of the delay between when changes are made and when those changes are applied.

Registration Timeout

Default Value: 5 s

Valid Values: Any positive integer with a valid unit of time (see [“Valid Units of Time”](#))

Changes Take Effect: Immediately

Used only when IVR Server is configured to perform agent login on a connected T-Server. If the DN on which the agent login is to be performed is successfully registered with the T-Server, IVR Server waits the specified amount of time before it checks again to determine whether the DN is now registered. After the DN is registered, the agent login is requested.

Retry Timeout

Default Value: 30 s

Valid Values: Any positive integer with a valid unit of time (see [“Valid Units of Time”](#))

Changes Take Effect: Immediately

If an error occurs in response to attempts to register a DN or log in an agent, IVR Server reattempts the specified operation after the amount of time.

Router Timeout

Default Value: 4 s

Valid Values: Any positive integer with a valid unit of time (see [“Valid Units of Time”](#))

Changes Take Effect: Immediately

Specifies the amount of time that IVR Server waits for routing or call treatment instructions before it performs default routing for the call.

Stat Timeout

Default Value: 3 s

Valid Values: Any positive integer with a valid unit of time (see “[Valid Units of Time](#)”)

Changes take Effect: Immediately

Specifies the amount of time that IVR Server waits for a reply from Stat Server for a noncached peek operation. After the timer expires, the last known good value is used.

Stop Waiting Timeout

Default Value: 30 s

Valid Values: Any positive integer with a valid unit of time (see “[Valid Units of Time](#)”)

Changes Take Effect: Immediately

Specifies the amount of time that IVR Server waits for the final stage of the two-stage release mechanism before it ends the call.

Unregister Timeout

Default Value: 3 s

Valid Values: Any positive integer with a valid unit of time (see “[Valid Units of Time](#)”)

Changes Take Effect: Immediately

Specifies the maximum amount of time that IVR Server waits for EventUnregistered from a connected T-Server when it attempts to unregister a DN. If this timer expires, IVR Server stops waiting for the reply and makes no further attempt to unregister the DN.

wait-for-ringing

Default Value: 1 min

Valid Values: Any positive integer and valid unit of time

Changes Take Effect: Immediately

Specifies the maximum time interval between receipt of a NewCall message from an IVR Driver and receipt of a corresponding EventRinging message for this call from the T-Server.

When the wait-for-ringing timer expires, IVR Server will clean up the call context associated with the NewCall request and reply to the IVR client application with an EndCall message and a timeout error indication.

In normal operating conditions this timer will never expire. However, if an IVR client application does not reliably send an EndCall message for every corresponding NewCall message, this timer can be used to clear the call context.

Note: This timer is applicable only for IVR Server in Behind mode.

TServerClientSap

This section must be called TServerClientSap.

call-timer-timeout

Default Value: 28800

Valid Values: Any valid integer

Changes Take Effect: Immediately

Specifies the amount of time (in seconds) that the TServerClientSap waits for the call to be released. After the timer expires, the TServerClientSap assumes that the call is over, and all call-related information is deleted.

defer-tlib-events

Default Value: true

Valid Values: true, false

Changes Take Effect: At I-Server startup

Specifies whether call-related events are processed before a new call notification for the call is received. If false, all call-related events are processed immediately. If true, all call-related events are buffered in the TServerClientSap until a new call notification is received for the call, at which time the events are processed.

Note: In order for I-Server to function properly, the defer-tlib-events option must be set to true.

waiting-for-newcall-timeout

Default Value: 30

Valid Values: Any valid integer

Changes Take Effect: Immediately

Specifies the amount of time (in seconds) that the TServerClientSap waits for a new call notification for a call. After the timer expires, the TServerClientSap assumes that the new call notification will never arrive, and all events for the call are discarded.

XmlSap

This section must be called `XmlSap`.

source-encoding

Default Value: Empty string

Valid Values: UTF-8

Changes Take Effect: At startup

When the option is set to UTF-8, client UTF-8 characters will be passed to T-Server in their UTF-8 format.

Note: Only UTF-8 characters of one or two bytes are supported.

target-encoding

Default Value: ISO-8859-1

Valid Values: ISO-8859-1, UTF-8

Changes Take Effect: At startup

Specifies the transcoder that is used to convert Unicode character data before user data is attached to a call. This transcoder is also used to convert user data back to Unicode when it is passed to the network.

Notes:

- Any keys or user data sent or received (including after the transcoding thereof) must be no more than 2 bytes per character.
- Only ISO-8859-1 and 2 byte UTF-8 characters are supported.

validation-scheme

Default Value: never

Valid Values: always, auto, never

Changes Take Effect: Immediately

Specifies whether validation is performed on incoming XML messages. If set to `never`, no validation of the XML messages is performed. If set to `auto`, XML messages are validated if a `.dtd` file exists; otherwise, no validation is performed. If set to `always`, all XML messages are validated, and it is considered an error if no `.dtd` file exists.

Note: Enable the `validation-scheme` option when you are performing integration testing for new client applications. Otherwise, Genesys recommends that you disable this option, because doing so improves performance.

I-Server Options

I-Server configuration options are defined in the `Options` of the I-Server `Application` object, using one of the following navigation paths:

- In Genesys Administrator—`Application` object > `Options` tab > `Advanced View (Options)`
- In Configuration Manager—`Application` object > `Properties` dialog box > `Options` tab

AgentLogin

This section must be called `AgentLogin`.

ReadyWorkMode

Default Value: `ManualIn`

Valid Values: `ManualIn`, `autoIn`, `Unknown`

Changes Take Effect: `Immediately`

Specifies the value that is used for `AttributeWorkMode` when IVR Server performs login operations. This value is used only in `TAgentReady`.

IgnoreReady

Default Value: `false`

Valid Values: `true`, `false`

Changes Take Effect: `Immediately`

Specifies whether I-Server attempts to change the agent's default ready state using `TAgentReady` or `TAgentNotReady` when it logs in that agent.

AgentLogout

This section must be called `AgentLogout`.

LogoutOnShutdown

Default Value: `true`

Valid Values: `true`, `false`

Changes Take Effect: `Immediately`

Specifies whether IVR Server logs out agents when it shuts down. If set to `true`, agents will be logged out at shutdown.

Note: Do not use the `LogoutOnShutdown` option if you are using the `app-name` option (see [page 245](#)).

TimePerLogout

Default Value: 100ms

Valid Values: Any positive integer

Changes Take Effect: Immediately

Specifies the maximum amount of time (in milliseconds) to wait for each successful logout before completing a shutdown.

LogoutOnDisable

Default Value: false

Valid Values: true, false

Changes Take Effect: Immediately

If set to true, any change to the configuration that renders a port unusable triggers agent logout. This applies only if the relevant port has login information configured, and if the server is operating in legacy mode.

InFront

This section must be called InFront.

UseQueue

Default Value: None

Valid Values: Any valid ACD queue name

Changes Take Effect: Immediately

Specifies the name of an ACD queue that has been associated in the Configuration Layer with the virtual switch specified for the IVR Server that is running in IVR-In-Front mode. All calls that the IVR Server receives are placed in this queue, which can be used with Inter Server Call Control (ISCC) direct-ani and dnis-pool external routing.

LoadBalance

This section must be called LoadBalance.

app-name

Default Value: None

Valid Values: Any valid I-Server name

Changes Take Effect: After TServer_IVR is restarted

Specifies the name of the primary I-Server in the Configuration Layer.

Stat:<stat name>

This section must be called Stat:<stat name>.

Note: In this section, <stat name> is the string that is passed in the XML messages when referencing this statistic.

obj_id

Default Value: None

Valid Values: Any valid object ID

Changes Take Effect: After I-Server is restarted

Specifies the object name, which is taken from Configuration Server (not database object identifier [DBID]).

For statistics of type SObjectGroupQueues, the format is GroupName@Tenant. For all other statistics, the format is DN@Switch. For the list of supported object types, see the description of the obj_type option.

obj_type

Default Value: None

Valid Values: SObjectRoutePoint, SObjectQueue, SObjectSwitch, SObjectRegDN, SObjectGroupQueues

Changes Take Effect: After I-Server is restarted

Specifies the object type of a statistic. You must configure these object types in Stat Server, using the syntax shown in the description of the obj_id option.

server_name

Default Value: None

Valid Values: Any valid Stat Server name

Changes Take Effect: After I-Server is restarted

Specifies the name of the Stat Server for statistic retrieval.

stat_type

Default Value: None

Valid Values: Any valid statistic type

Changes Take Effect: After I-Server is restarted

Specifies the type of statistic that is to be requested. These stat types must be configured in Stat Server.

time_profile

Default Value: None

Valid Values: Any valid time profile that is defined in Stat Server

Changes Take Effect: After I-Server is restarted

Specifies the time profile to be used for the statistic.

update_frequency

Default Value: No default value

Valid Values: Any integer

Changes Take Effect: After I-Server is restarted

Specifies the update frequency for the statistic. If ≤ 0 , no updates from Stat Server are requested, and each request to I-Server for statistic information causes I-Server to send a request to Stat Server. If > 0 , Stat Server sends statistic updates to I-Server, based on the value (in seconds) that is set in the update_frequency option. A request to I-Server for statistic information causes I-Server to return the cached value.

VirtualRoutePoints

This section must be called VirtualRoutePoints.

<Premise T-Server application name in CME>

Default Value: None

Valid Values: See Table 16 on [page 248](#)

Changes Take Effect: Immediately

Specifies the available Virtual Routing Points for route requests in an IVR-Behind-Switch configuration.

To set this option:

1. From within the VirtualRoutePoints section of an I-Server application, click **Create New Section/Option**. The **Edit Option** dialog box appears.
2. In the **Option Name** box, specify the exact name of a premise T-Server as it is configured in the Configuration Layer.
3. In the **Option Value** box, specify one or more values, using the syntax shown in [Table 16](#).
4. Click **OK** to create the option.

Note: If the number of configured Virtual Routing Points exceeds 255, you must segment the Virtual Routing Points that are defined for a given I-Server (on its Options tab), by using the ExtensionDelimiter option.

Table 16: Valid Values for <Premise T-Server Application Name in CME>

Configuration Conditions	Valid Values	Variable Definitions
Using Load Balancing	DN1[aliasname]:DN2[aliasname]	DN1–DN <i>n</i> are configured Virtual Routing Points on the premise T-Server's switch.
Not using Load Balancing	DN1:DN2:DN <i>n</i>	
Using more than 255 Virtual Routing Points	<premise T-Server name><ExtensionDelimiter> <ID> = VRP1[aliasname]:VRP2[aliasname]...	<ID> can be any string (such as 001, 002, and so on).

ExtensionDelimiter

Default Value: : (colon)

Valid Values: Any nonzero, single-character string

Changes Take Effect: Immediately

Used as a token to parse extended Virtual Routing Points definitions when the list exceeds 255 entries.

IVR_Driver Options

IVR_Driver configuration options are set in the Options of the IVR_Driver Application object, using one of the following navigation paths:

- In Genesys Administrator—Application object > Options tab > Advanced View (Options)
- In Configuration Manager—Application object > Properties dialog box > Options tab.

Note: All options and values other than those that are described in this section are ignored.

ivr_server_interface

This section must be called `ivr_server_interface`.

callinfo_all_returns_3rdpartydnDefault Value: `false`Valid Value: `true`, `false`

Changes Take Effect: Immediately

Specifies what response data will be returned to the client application for a `GetCallInfo (All)` message.

If set to `true`, a `GetCallInfo (All)` response message will contain the current value for `ThirdPartyDN`.

If set to `false`, `ThirdPartyDN` will not be contained within the `GetCallInfo (All)` response message.

callinfo_all_returns_uuid

Default Value: `false`

Valid Value: `true`, `false`

Changes Take Effect: Immediately

Specifies what response data is returned to the client application for a `GetCallInfo (All)` message.

If set to `true`, a `GetCallInfo (All)` response message will contain the current value for `UUID`.

If set to `false`, `UUID` will not be contained within the `GetCallInfo (All)` response message.

compat65

Default Value: `false`

Valid Value: `true`, `false`

Changes Take Effect: Immediately

Specifies what response data will be returned to the driver application for a `GetCallInfo` response message or a `UData` response message.

If any value in a `GetCallInfo` response message is null and `compat65` is set to `true`, the response data will contain `Not Available`. Otherwise, it will contain `NULL`.

For a `UData` response message, if the response is not successful and `compat65` is set to `true`, the response data will contain `Error NoMatch`. Otherwise, it will contain `NoMatch`.

getreply_with_location

Default Value: `false`

Valid Value: `true`, `false`

Changes Take Effect: Immediately

In some responses, which have the `DEST` element present, there will also be a `SWITCH` key-value pair. For these cases, if `getreply_with_location` is set to `true`, the value returned in the `RouteResponse` message will be of the form `DN@SWITCH`.

iserver_mode_hotstandby

Default Value: `false`

Valid Values: `true`, `false`

Changes Take Effect: After IVR Driver restart

If the value is `true`, this indicates that the IVR Servers are configured in Hot Standby mode. It is expected that two and only two IVR Servers are in the `load_sharing_iservers` list. Communications with the IVR Servers will be done to facilitate hot standby. All request messages are sent to both configured IVR Servers to enable their synchronization. The response to the request message is returned by only the primary IVR Server. If the IVR Servers are not configured for Hot Standby mode, the results are unpredictable.

load_sharing_iservers

Default Value: None

Valid Values: A comma-delimited list of (host:port) pairs for all IVR Servers to which the IVR Driver connects

Changes Take Effect: Immediately

Specifies a comma-delimited list of (host:port) pairs of Load Sharing I-Server applications.

Note: Spaces are not permitted anywhere in comma-delimited lists.

load_sharing_iservers_client_hosts

Default Value: `false`

Valid Value: `true`, `false`

Changes Take Effect: Immediately

Specifies whether to use the host address in the transport section when connecting to IVR Server.

load_sharing_iservers_client_ports

Default Value: No default value

Valid Value: Comma-delimited list of port numbers

Changes Take Effect: Immediately

Specifies a comma-delimited list of ports to use to connect to IVR Server. The number of entries must match the number of load-sharing IVR Servers.

socket_activity_timer

Default Value: `20000`

Valid Values: Any integer ≥ 1000 , or `0`

Changes Take Effect: Immediately

Specifies the amount of time (in milliseconds) before IVR Library suspects that the socket connection to IVR Server is disconnected.

If IVR Library does not receive any event messages from IVR Server within the amount of time specified by `socket_activity_timer`, IVR Library sends a `KeepAlive` message to IVR Server. If IVR Library does not receive any events from IVR Server within three times the `socket_activity_timer` value, it disconnects the socket and attempts to open a new one.

If set to 0, `KeepAlive` processing is disabled.

time_recon_is

Default Value: 2000

Valid Value: Any integer ≥ 1000

Changes Take Effect: Immediately

Specifies the amount of time (in milliseconds) between reconnection attempts.

log_content

This section must be called `log_content`.

Note: The options in the `log_content` section apply only to debug-level messages. The other logging options that you can set for the `IVR_Driver` application are documented in Chapter 11, “Common Configuration Options,” on [page 209](#).

log_print_agent_login

Default Value: true

Valid Values: true, false

Changes Take Effect: Immediately

Specifies whether to print the agent login message which contains the agent password.

log_print_date

Default Value: yes

Valid Values: yes, no

Changes Take Effect: Immediately

Specifies whether to include the date stamp in output.

log_print_driver_selector

Default Value: 0

Valid Values: A string of zeros and ones

Changes Take Effect: Immediately

Specifies which driver messages are logged.

log_print_hb

Default Value: no

Valid Values: yes, no

Changes Take Effect: Immediately

Specifies whether to log heartbeat requests in output.

log_print_level

Default Value: xml

Valid Values: flow, xml, debug, detail, none

Changes Take Effect: Immediately

Specifies the level of logging:

- `flow`—Logs the messages flowing from the application to IVR Library to IVR Server, and then back.
- `xml`—Logs all messages provided by `flow`, as well as the XML strings that flow from IVR Library to IVR Server, and then back.
- `debug`—Logs all messages provided by `xml`, as well as internal debug messages.
- `detail`—Logs all messages provided by `debug`, as well as internal debug flow messages.
- `none`—Logs all messages, as specified in the verbose option of the `log` section.

log_print_login_requests

Default Value: yes

Valid Values: yes, no

Changes Take Effect: Immediately

Specifies whether to log Login Request XML strings.

log_print_name

Default Value: yes

Valid Values: yes, no

Changes Take Effect: Immediately

Specifies whether to include the IVR Driver name in output.

log_print_recv

Default Value: yes

Valid Values: yes, no

Changes Take Effect: Immediately

Specifies whether to log messages that IVR Server sends to the IVR Driver.

log_print_send

Default Value: yes

Valid Values: yes, no

Changes Take Effect: Immediately

Specifies whether to log messages sent by the driver to IVR Server.

log_print_time

Default Value: yes

Valid Values: yes, no

Changes Take Effect: Immediately

Specifies whether to include the time (in hrs:min:sec format) in output.

log_print_time_ms

Default Value: yes

Valid Values: yes, no

Changes Take Effect: Immediately

Specifies whether to include the time (in hrs:min:sec:ms format) in output. If the log_print_time option is set to No, this option is ignored.

log_print_timeouts

Default Value: no

Valid Values: yes, no

Changes Take Effect: Immediately

Specifies whether to print IVR Driver (GetReply) timeouts to the log.

log_print_udata

Default Value: no

Valid Values: yes, no

Changes Take Effect: Immediately

Specifies whether to include in output the contents of the value in key-value data pairs.

IVR Driver Annex Options

Options in this section are defined in the Annex of the Application object, as follows:

- in Genesys Administrator—Application object > Options tab > Advanced View (Annex)
- in Configuration Manager—Application object > Properties dialog box > Annex tab

DriverIgnoreReady

Default Value: false

Valid Values: true, false

Changes Take Effect After IVR Driver restart

Specifies whether to ignore the setReady value which is configured for an agent when the agent is logged in.

DriverReadyWorkMode

Default Value: ManualIn

Valid Values: ManualIn, AutoIn, Unknown

Changes Take Effect After IVR Driver restart

If you attempt to go NotReady on some switches, like G3R, and there is a call on the port, the switch will not allow this and will send an error message. In order to do graceful shutdown on a G3R (and possibly others), when the Ready message is sent to IServer, the workmode parameter must be set to ManualIn. This causes the switch to make the agent NotReady when the call he or she is on ends. Because of this, first check to see if ManualIn is set and, if not, send the NotReady message. If it is set, then check to see if a call is in progress. If it is not, send the NotReady message. If there is a call in progress, wait for the switch to send a NotReady message.

DriverRetryTimeout

Default Value: 60ms

Valid Values: Any integer value > 1

Changes Take Effect: After IVR Driver restart

This parameter is used when errors occur during attempts to process agent control messages. If an agent control message attempting to log out an agent fails, you should make the assumption that the agent is logged out. In this case the DriverRetryTimeout is not used. If any agent control message attempting either to query an agent state or log in, make ready, or make not ready fails, set the agent state to unknown and set a timer equal to the value of DriverRetryTimeout. Wait until the timer is exceeded and run a query on the agent to see what state it is in. If the query succeeds, the state is known, attempt to bring it to its configured state. If the query fails, this cycle will repeat.

IVR Annex Options

Options in this section are defined in the Annex of the Application object, as follows:

- in Genesys Administrator—Application object > Options tab > Advanced View (Annex)
- in Configuration Manager—Application object > Properties dialog box > Annex tab

AgentControl

The option in this section is used to specify which AgentControl values IVR Library expects and what effect they have.

Note: All values other than those that are described in this section are ignored.

LegacyMode

Default Value: true

Valid Values: true, false

Changes Take Effect: Immediately

Specifies whether IVR Server or IVR Driver controls agent state, in order to provide consistency with the values that are configured for the IVR ports in the IVR object (see “AgentLogin” on [page 244](#)). If set to true, the IVR Server controls the agent activity. If set to false, the IVR Driver or the IVR SDK controls the agent activity.

DataTransport

The options in this section are used to specify which DataTransport values IVR Library expects, and what effect they have.

Note: All values other than those that are described in this section are ignored.

Starting with IVR Server 7, if you are using the IVR Drivers for Aspect and CONVERSANT, or the IVR SDK, you must configure these options on the Annex tab of the IVR Properties dialog box. For all other IVR Drivers, there are corresponding options that you configure on the Options tab of the IVR Properties dialog box. See “IVR_Driver Options” on [page 248](#).

log_file_name

Default Value: con

Valid Values: Any valid file name

Changes Take Effect: After the IVR Driver is restarted

Specifies the name of the log file, including the path (if desired). The IVR Driver must have write access to the path, and all directories along the path must already be created.

log_file_size

Default Value: 0

Valid Values: Any integer

Changes Take Effect: After the IVR Driver is restarted

Specifies the maximum size (in bytes) of log files. If empty or set to 0, no limit is implied.

log_file_backup_amount

Default Value: 0

Valid Values: Any integer

Changes Take Effect: After the IVR Driver is restarted

Specifies the number of backup log files.

time_recon_is

Default Value: 2000

Valid Values: Any integer \geq to 1000

Changes Take Effect: After the IVR Driver is restarted

Specifies the amount of time (in milliseconds) between reconnection attempts.

log_print_timeouts

Default Value: no

Valid Values: yes, no

Changes Take Effect: After the IVR Driver is restarted

Specifies whether to print Driver(GetReply) timeouts to the log.

log_print_name

Default Value: yes

Valid Values: yes, no

Changes Take Effect: After the IVR Driver is restarted

Specifies whether to include the IVR Driver name in output.

log_print_date

Default Value: `yes`

Valid Values: `yes`, `no`

Changes Take Effect: After the IVR Driver is restarted

Specifies whether to include the date stamp in output.

log_print_time

Default Value: `yes`

Valid Values: `yes`, `no`

Changes Take Effect: After the IVR Driver is restarted

Specifies whether to include the time (in `hrs:min:sec` format) in output.

log_print_time_ms

Default Value: `yes`

Valid Values: `yes`, `no`

Changes Take Effect: After the IVR Driver is restarted

Specifies whether to include the time (in `hrs:min:sec:ms` format) in output. If the `log_print_time` option is set to `No`, this option is ignored.

log_print_udata

Default Value: `no`

Valid Values: `yes`, `no`

Changes Take Effect: After the IVR Driver is restarted

Specifies whether to include in output the contents of the value in key-value data pairs.

log_print_level

Default Value: `xml`

Valid Values: `flow`, `xml`, `debug`, `detail`, `none`

Changes Take Effect: After the IVR Driver is restarted

Specifies the level of logging:

- `flow`—Logs the messages flowing from the application to IVR Library to IVR Server, and then back.
- `xml`—Logs all messages provided by `flow`, as well as the XML strings flowing from IVR Library to IVR Server, and then back.
- `debug`—Logs all messages provided by `xml`, as well as internal debug messages.
- `detail`—Logs all messages provided by `debug`, as well as internal debug flow messages.
- `none`—Logs all messages, as specified in the `verbose` option of the `log` section.

log_print_hb

Default Value: no

Valid Values: yes, no

Changes Take Effect: After the IVR Driver is restarted

Specifies whether to log heartbeat requests in output.

log_print_send

Default Value: yes

Valid Values: yes, no

Changes Take Effect: After the IVR Driver is restarted

Specifies whether to log messages that the IVR Driver sends to IVR Server.

load_sharing_1servers

Default Value: None

Valid Values: A comma-delimited list of (host:port) pairs of Load Sharing I-Servers

Changes Take Effect: After the IVR Driver is restarted

Specifies a comma-delimited list of (host:port) pairs of Load Sharing I-Servers.

log_print_recv

Default Value: yes

Valid Values: yes, no

Changes Take Effect: After the IVR Driver is restarted

Specifies whether to log messages that IVR Server sends to the IVR Driver.

log_print_login_requests

Default Value: yes

Valid Values: yes, no

Changes Take Effect: After the driver is restarted.

Specifies whether to log Login Request XML strings.

log_print_driver_selector

Default Value: 0 (zero)

Valid Values: A string of zeroes and '1's

Changes Take Effect: After the driver is restarted.

Specifies which driver messages are logged.

socket_activity_timer

Default Value: 20000

Valid Values: Any integer ≥ 1000 , or 0

Changes Take Effect: Immediately

Specifies the amount of time (in milliseconds) before IVR Library suspects that the socket connection to IVR Server is disconnected.

If IVR Library does not receive any event messages from IVR Server within the amount of time specified by `socket_activity_timer`, IVR Library sends a `KeepAlive` message to IVR Server. If IVR Library does not receive any events from IVR Server within three times the `socket_activity_timer` value, it disconnects the socket and attempts to open a new one.

If set to 0, `KeepAlive` processing is disabled.

IVR Port Annex Options

Options in this section are defined in the Annex of the IVR Port Properties object, as follows:

- in Genesys Administrator—Application object > Options tab > Advanced View (Annex)
- in Configuration Manager—Application object > Properties dialog box > Annex tab

AutoLogin

This section must be called `AutoLogin`.

AgentId

Default Value: None

Valid Values: Any valid AgentID

Changes Take Effect: Immediately

Specifies the AgentID that is related to the IVR port.

Queue

Default Value: None

Valid Values: Any valid queue

Changes Take Effect: Immediately

Specifies the queue to log in to.

Password

Default Value: ""

Valid Values: Any valid password

Changes Take Effect: Immediately

Specifies the password for the specified AgentID.

SetReady

Default Value: false

Valid Values: true, false

Changes Take Effect: Immediately

Specifies which message to send after the agent logs in. If set to true, a RequestSetAgentReady message is sent. If set to false, a RequestSetAgentNotReady message is sent.

SetLoggedIn

Default Value: true

Valid Values: true, false

Changes Take Effect: Immediately

Specifies whether an agent should be logged in.

Changes from Release 8.0 to 8.1

Table 17 lists the configuration options that:

- Are new or changed in the 8.1 release of IVR Server
- Have been added or changed since the most recent 8.0 release of this document

If a configuration option has been replaced with another that enables the same functionality, the new option name and its location in this chapter are noted.

Table 17: Option Changes from Release 8.0 to 8.1

Option Name	Option Values	Type of Change	Details
TServer_IVR: XmlSap Section			
source-encoding	true, false	New in 8.0	See the option description on page 243 .
TServer_IVR: pgf-debug Section			
debug	default:ALL	Not previously documented	See the option description on page 239 .

Table 17: Option Changes from Release 8.0 to 8.1 (Continued)

Option Name	Option Values	Type of Change	Details
hide-user-data	true, false	New in 8.1	See the option description on page 239 .
IVR Driver: ivr_server_interface Section			
callinfo_all_returns_3rdpartydn	true, false	New in 8.1	See the option description on page 248 .
callinfo_all_returns_uuid	true, false	New in 8.1	See the option description on page 248 .

A

Sample Configurations

This appendix provides examples of how to configure Interactive Voice Response devices (IVRs) in each of the three configuration modes (IVR-In-Front, IVR-Behind-Switch, and IVR Network T-Server), using Configuration Manager. It contains the following sections:

- [IVR-In-Front Configuration, page 263](#)
- [IVR-Behind-Switch Configuration, page 270](#)
- [IVR Network T-Server Configuration, page 275](#)

Note: Before you configure IVR Server, make sure that the Annex tab is enabled on all applicable **Properties** dialog boxes. For instructions, see “Enabling the Annex Tab” on [page 141](#).

IVR-In-Front Configuration

[Figure 16](#) illustrates a sample IVR-In-Front configuration.

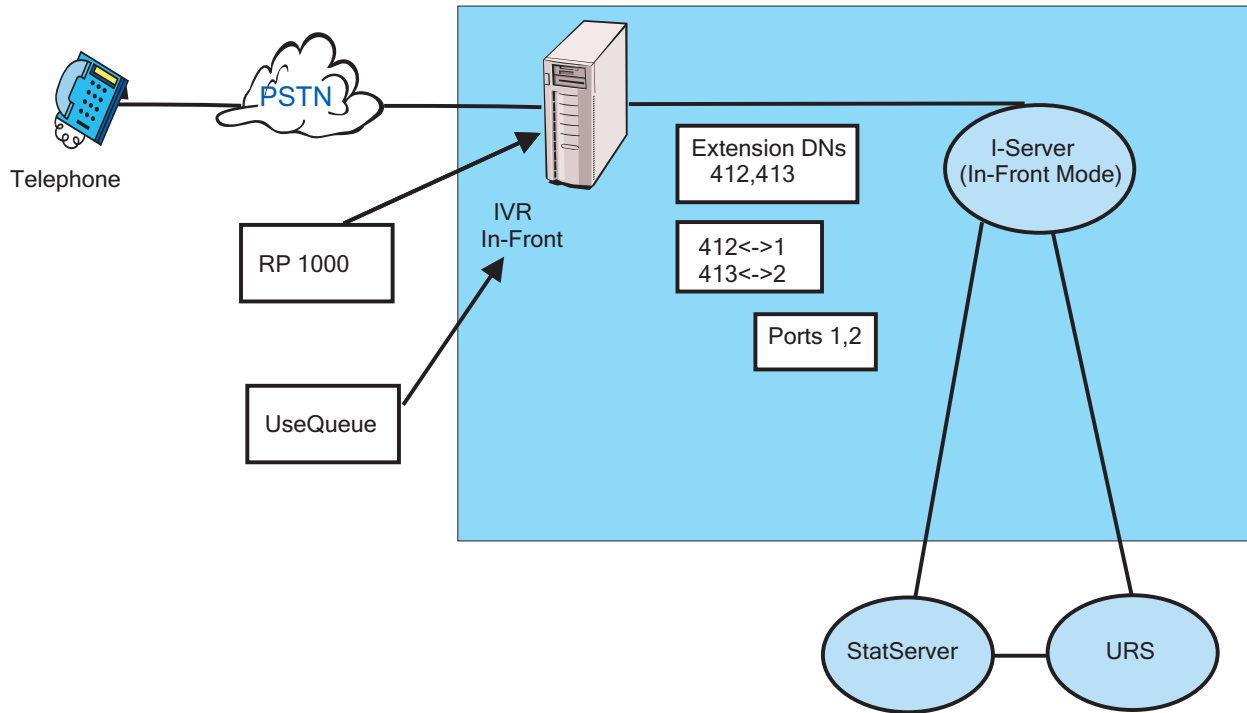


Figure 16: IVR-In-Front Sample Configuration

In this sample IVR-In-Front configuration, the IVR has two ports: 1 and 2. These ports are mapped to DNs 412 and 413 on the virtual switch. In addition, the virtual switch has defined Routing Point 1000 to be used for route requests. This example also shows statistics, routing services (through Universal Routing Server [URS]), and Load Sharing of IVR Server options are also shown.

The procedures in this section explain the following steps in detail:

1. Configure the switching office.
2. Configure the switch.
 - Configure the DNs.
3. Configure the IVR object.
 - Configure the IVR ports.
 - Configure the agent login (optional).
4. Configure the I-Server application.
 - Configure the ACD queue for the virtual switch (optional).
5. Configure the TServer_IVR application.
6. Associate the IVR object with the I-Server application.

7. Connect the TServer_IVR and Stat Server applications to the I-Server application.

Note: If you want to create routing strategies, you must configure a connection from the URS application to the TServer_IVR application.

Configuring the Switching Office

To configure the IVR-In-Front switching office in Configuration Manager:

1. Expand Environment, right-click Switching Offices, and select New > Switching Office. The New Switching Office Properties dialog box appears.
2. In the Name box, enter IVR In-Front Switching Office.
3. In the Switch Type box, select Virtual Switch for IVR In-Front.
4. Click OK to complete the configuration of the IVR-In-Front switching office.

Configuring the Switch

To configure the IVR-In-Front switch in Configuration Manager:

1. Expand Resources (or Tenant Name), right-click Switches, and select New > Switch. The IVR In-Front Switch Properties dialog box appears.
2. In the Name box, enter IVR In-Front Switch.
3. Click the Folder icon next to the Switching Office box, select IVR In-Front Switching Office, and click OK.
4. Click OK to complete the configuration of the IVR-In-Front switch.

Configuring the DNs

To configure DNs 412 and 413, and Routing Point 1000 in Configuration Manager:

1. Expand Resources (or Tenant Name) > Switches > IVR In-Front Switch, right-click DNs, and select New > DN. The New DN Properties dialog box appears.
2. Configure DN 412:
 - a. In the Number box, enter 412.
 - b. In the Type box, select Extension.
 - c. Click Apply.
3. Configure DN 413:
 - a. In the Number box, enter 413.

- b. In the Type box, select Extension.
 - c. Click Apply.
- 4. Configure DN 412:
 - a. In the Number box, enter 1000.
 - b. In the Type box, select Routing Point.
 - c. Click OK.

Configuring the IVR

To configure the IVR in Configuration Manager:

1. Expand Resources (or Tenant Name), right-click IVRs, and select New > IVR. The New IVR Properties dialog box appears.
2. In the Name box, enter In-Front IVR.
3. In the Type box, select the applicable IVR type.
4. In the Version box, enter the applicable IVR version.
5. Click the Folder icon next to the IVR Interface Server box, select I-Server_810, and click OK.
6. Click the Annex tab, and do the following:
 - a. Create a new section named DataTransport.
 - b. Enter the key-value pairs that are to be returned when users log in to IVR Server.

You can accept the defaults for the other options on the Annex tab, unless you need specific option definitions.

Note: Step 6 applies only to the IVR Drivers 7.5 for Aspect Unixware and CONVERSANT. For other IVR Drivers, skip this step.

7. Click OK to complete the configuration of the IVR.

Configuring the IVR Ports

To configure the IVR ports in Configuration Manager:

1. Expand Resources (or Tenant Name) > IVRs > In-Front IVR, right-click IVR Ports, and select New > IVR Port. The New IVR Port Properties dialog box appears.
2. Configure Port 1:
 - a. In the Port Number box, enter 1.
 - b. Click the Folder icon next to the Associated DN box, select DN 412 for the associated switch (IVR In-Front Switch), and click OK.
 - c. Click Apply.

3. Configure Port 2:
 - a. In the Port Number box, enter 2.
 - b. Click the Folder icon next to the Associated DN box, select DN 413 for the associated switch (IVR In-Front Switch), and click OK.
 - c. If you are using the Auto-Login feature, click the Annex tab and do the following:
 - Create a new section named AutoLogin.
 - Add configuration options for the AutoLogin section, as described in “Configuring the Auto-Login Feature” on [page 150](#), and “AgentLogin” on [page 244](#).
 - d. Click OK.

Creating and Configuring the I-Server Application

To create and configure an I-Server application named IVR Server, of type IVR Interface Server, in Configuration Manager:

1. Expand Environment, right-click Applications, and select New > Application. The Browse dialog box appears.
2. Select I-Server_810, and then click OK. The New I-Server_810 Properties dialog box appears.
3. In the Name box, enter IVR Server.
4. Click Apply to save your changes on this tab.
5. Click the Start Info tab.
6. Enter a single period (.) in the following boxes:
 - Working Directory
 - Command Line
 - Command Line Arguments
7. Click Apply to save your changes on this tab.
8. Click the Server Info tab.
9. In the Host box, enter the host that the I-Server application will use.
10. In the Ports box, add a new port number that the I-Server application will use.
11. Click Apply to save your changes on this tab.
12. Click the Options tab.
13. If you want to use Load Balancing:
 - a. Create a new section named LoadBalance.
 - b. Add configuration options, as described in “LoadBalance” on [page 245](#).
14. If you want to gather statistics:
 - a. Create a new section named Stat:<stat name>.

16. Create the following new sections (if they have not already been defined):

- | | |
|----------------------|--------------------|
| • CallIdSap | • IServerGLMSap |
| • extrouter | • license |
| • gli | • log |
| • gli_server | • Timers |
| • gli_server_group_1 | • TServer |
| • IServer | • TServerClientSap |

17. You must define (or add) the following options:

- gli_server_address in the gli_server_group_1 section.
- app_name in the IServer section.
- operation-mode in the IServerGLMSap section.
- license-file in the license section..

Note: In most cases, you can use the default values for the other options. For option descriptions, see Chapter 12 on [page 231](#).

18. Click Apply to save your changes on this tab.

19. Click OK to complete the configuration of the TServer_IVR application.

Associating the IVR with the I-Server Application

To connect the IVR to the I-Server application in Configuration Manager:

1. Expand Resources (or Tenant Name) > IVR, right-click In-Front IVR, and select Properties. The In-Front IVR Properties dialog box appears.
2. Click the Folder icon next to the IVR Interface Server box, select I-Server_810, and click OK.
3. Click OK to associate the IVR with the I-Server application.

Connecting the TServer_IVR and Stat Server Applications to the I-Server Application

To connect the TServer_IVR and Stat Server applications to the I-Server application in Configuration Manager:

1. Expand Environment > Applications, right-click I-Server_810, and select Properties. The I-Server_810 Properties dialog box appears.
2. Click the Connections tab.
3. Click Add to add connections to the:
 - TServer_IVR application
 - Stat Server (if required)
 - Message Server
4. Click OK to save your connections.

IVR-Behind-Switch Configuration

Figure 17 illustrates a sample IVR-Behind-Switch configuration.

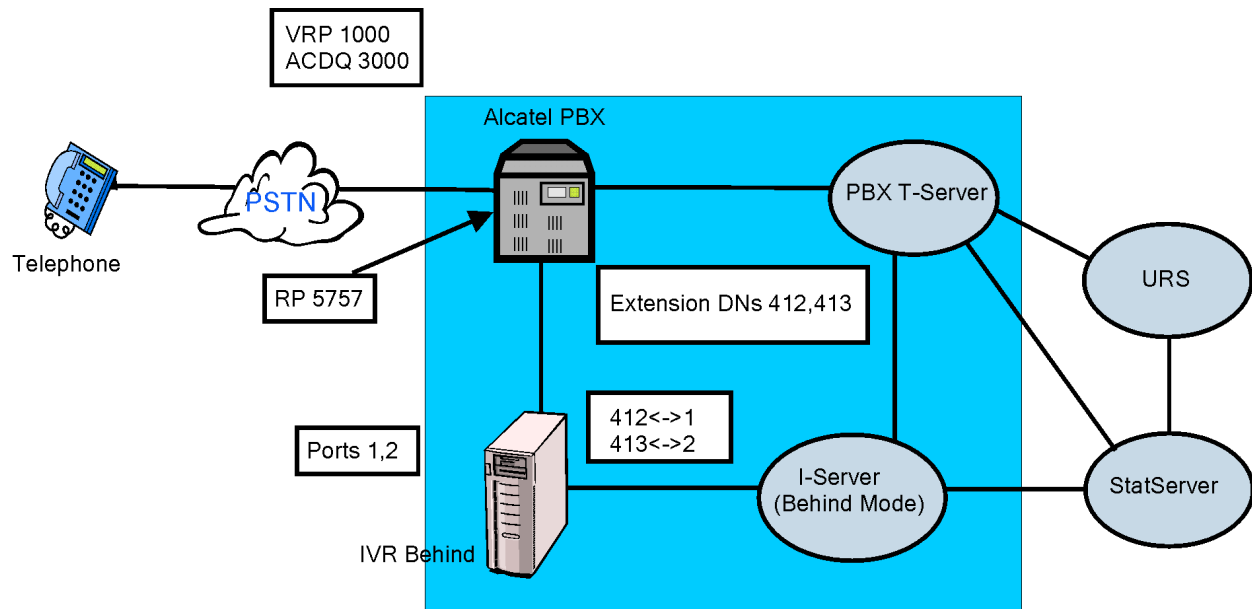


Figure 17: IVR-Behind-Switch Sample Configuration

Assume that premise T-Servers (PBX T-Server) have been configured, along with the required DNs (412 and 413), the Virtual Routing Point (1000), the switch (IVR-Behind-Switch switch), and the switching office (IVR-Behind-Switch switching office).

The procedures in this section explain the following steps in detail:

1. Configure the switching office.
2. Configure the switch.
3. Configure the IVR object.
4. Configure the IVR ports.
5. Define ACE Queue 3000.
6. Define the Routing Point DN.
7. Configure the I-Server application.
8. Configure the TServer_IVR application.
9. Associate the IVR object with the I-Server application.

10. Connect the I-Server application to other Genesys applications.

Note: The IVR-Behind-Switch configuration uses a virtual switch and a virtual switching office, in addition to the premise switch configuration described in the following sections. If you have not already configured a virtual switch and a virtual switching office, see “Creating a virtual switching office” on [page 115](#) and “Creating a virtual switch” on [page 116](#) for information on how to configure them.

Configuring the Switching Office

To configure the IVR-Behind-Switch switching office in Configuration Manager:

1. Expand Environment, right-click Switching Offices, and select New > Switching Office. The New Switching Office Properties dialog box appears.
2. In the Name box, enter IVR Behind Switching Office.
3. In the Switch Type box, select Virtual Switch for IVR In-Front.
4. Click OK to complete the configuration of the IVR-Behind-Switch switching office.

Configuring the Switch

Note: Before you complete the following steps, you must configure a switching office, as described in “[Configuring the Switching Office.](#)”

To configure the IVR-Behind-Switch switch in Configuration Manager:

1. Expand Resources (or Tenant Name), right-click Switches, and select New > Switch. The New Switch Properties dialog box appears.
2. In the Name box, enter IVR T-Server Switch.
3. Click the Folder icon next to the Switching Office box, select IVR Behind Switching Office, and click OK.
4. Click OK to complete the configuration of the IVR-Behind-Switch switch.

Configuring the IVR

To configure the IVR in Configuration Manager:

1. Expand Resources (or Tenant Name), right-click IVRs, and select New > IVR. The New IVR Properties dialog box appears.
2. In the Name box, enter Behind IVR.

3. In the **Type** box, select the applicable IVR type.
4. In the **Version** box, enter the applicable IVR version.
5. Click the **Annex** tab, and do the following:
 - a. Create a new section named **DataTransport**.
 - b. Enter the key-value pairs that are to be returned when users log in to IVR Server.

You can accept the default values for the other options on the **Annex** tab, unless you need specific option definitions.

Note: [Step 5](#) applies only to the IVR Drivers 7.5 for Aspect Unixware and CONVERSANT. For other IVR Drivers, skip this step.

6. Click **OK** to complete the configuration of the IVR.

Configuring the IVR Ports

To configure the IVR ports in Configuration Manager:

1. Expand **Resources (or Tenant Name) > IVRs > Behind IVR**, right-click **IVR Ports**, and select **New > IVR Port**. The **New IVR Port Properties** dialog box appears.
2. Configure Port 1:
 - a. In the **Port Number** box, enter 1.
 - b. Click the **Folder** icon next to the **Associated DN** box, select **DN 412** for the associated switch (**IVR T-Server Switch**), and click **OK**.
 - c. Click **Apply**.
3. Configure Port 2:
 - a. In the **Port Number** box, enter 2.
 - b. Click the **Folder** icon next to the **Associated DN** box, select **DN 413** for the associated switch (**IVR T-Server Switch**), and click **OK**.
 - c. If you are using the **Auto-Login** feature, click the **Annex** tab and do the following:
 - Create a new section named **AutoLogin**.
 - Add configuration options for the **AutoLogin** section, as described in “Configuring the Auto-Login Feature” on [page 150](#), and “AgentLogin” on [page 244](#).
 - d. Click **OK**.

Defining ACD Queue 3000

To define ACD Queue 3000 in Configuration Manager:

1. Select **File > New > DN**.

2. In the Number box, enter 3000.
3. In the Type box, select ACD Queue.
4. Click OK to complete the definition of ACD Queue 3000.

Defining the Virtual Routing Point DN

To define Routing Point 1000 in Configuration Manager:

1. Expand Resources (or Tenant Name) > Switches > [Premise T-Server Switch], right-click DNs, and select New > DN. The New DN Properties dialog box appears.
2. In the Number box, enter 1000.
3. In the Type box, select Virtual Routing Point.
4. In the Register box, select False.
5. Click OK to complete the definition of Routing Point 1000.

Configuring the I-Server Application

To configure the I-Server application in Configuration Manager:

1. Expand Environment, right-click Applications, and select New > Application. The Browse dialog box appears.
2. Select I-Server_810, and then click OK. The New I-Server_810 Properties dialog box appears.
3. In the Name box, enter IVR Server.
4. Click Apply to save your changes on this tab.
5. Click the Start Info tab.
6. Enter a single period (.) in the following boxes:
 - Working Directory
 - Command Line
 - Command Line Arguments
7. Click Apply to save your changes on this tab.
8. Click the Options tab.
9. If you want to use Load Balancing:
 - a. Create a new section named LoadBalance.
 - b. Add configuration options, as described in “LoadBalance” on [page 245](#).
10. If you want to gather statistics:
 - a. Create a new section named Stat:<stat name>.
 - b. Add configuration options, as described in “Stat:<stat name>” on [page 246](#).

11. If you want to use routing:
 - a. Create a new section named `VirtualRoutePoints`.
 - b. Add the value `PBX T-Server=1000`.
12. If you want to use automatic agent logout:
 - a. Create a new section named `AgentLogout`.
 - b. Add configuration settings for the `LogoutOnShutdown` and `TimePerLogout` options, as described in “AgentLogout” on [page 244](#).
13. Click Apply to save your changes on this tab.
14. Click OK to configure the I-Server application.

Configuring the TServer_IVR Application

To configure the TServer_IVR application in Configuration Manager:

1. Expand Environment, right-click Applications, and select New > Application. The Browse dialog box appears.
2. Select TServer_IVR_810, and then click OK. The New TServer_IVR_810 Properties dialog box appears.
3. In the Name box, enter IVR T-Server.
4. Click Apply to save your changes on this tab.
5. Click the Start Info tab.
6. In the Working Directory and Command Line boxes, enter the applicable information, as described in [Steps 17–19 on page 160](#).
7. Click Apply to save your changes on this tab.
8. Click the Server Info tab.
9. In the Host box, enter the host that the TServer_IVR application will use.
10. In the Ports box, add a new port number that the TServer_IVR application will use.
11. Click Apply to save your changes on this tab.
12. Click the Options tab.

Note: The following sections and configuration options are created in the template, but some must be modified.

13. Create the following new sections (if they have not already been defined):

• CallIdSap	• IServerGLMSap
• extrouter	• license
• gli	• log
• gli_server	• Timers
• gli_server_group_1	• TServer
• IServer	• TServerClientSap

14. You must define (or add) the following options:

- `gli_server_address` in the `gli_server_group_1` section.
- `app_name` in the `IServer` section.
- `operation-mode` in the `IServerGLMSap` section.
- `license-file` in the `License` section.

Note: In most cases, you can accept the default values for the other options. For option descriptions, see Chapter 12 on [page 231](#).

15. Click Apply to save your changes on this tab.

16. Click OK to complete the configuration of the `TServer_IVR` application.

Associating the IVR with the I-Server Application

To connect the IVR to the I-Server application in Configuration Manager:

1. Expand Resources (or Tenant Name) > IVRs, right-click Behind IVR, and select Properties. The Behind IVR Properties dialog box appears.
2. In the IVR Interface Server box, select IVR Server.
3. Click OK to connect the IVR to the I-Server application.

Connecting the I-Server Application to Other Applications

To connect the I-Server application to the `TServer_IVR`, premise T-Server, Message Server, and Stat Server applications in Configuration Manager:

1. Expand Environment > Applications, right-click I-Server, and select Properties. The I-Server_810 Properties dialog box appears.
2. Click the Connections tab.
3. Click Add to add connections to the following:
 - `TServer_IVR` application
 - Premise T-Server application
 - Stat Server (if required)
 - Message Server (if required)
4. Click OK to save your connections.

IVR Network T-Server Configuration

Note: For information about how to configure the Network T-Server, see the *Deployment Guide* for your Network T-Server.

Figure 18 illustrates a sample IVR Network T-Server configuration.

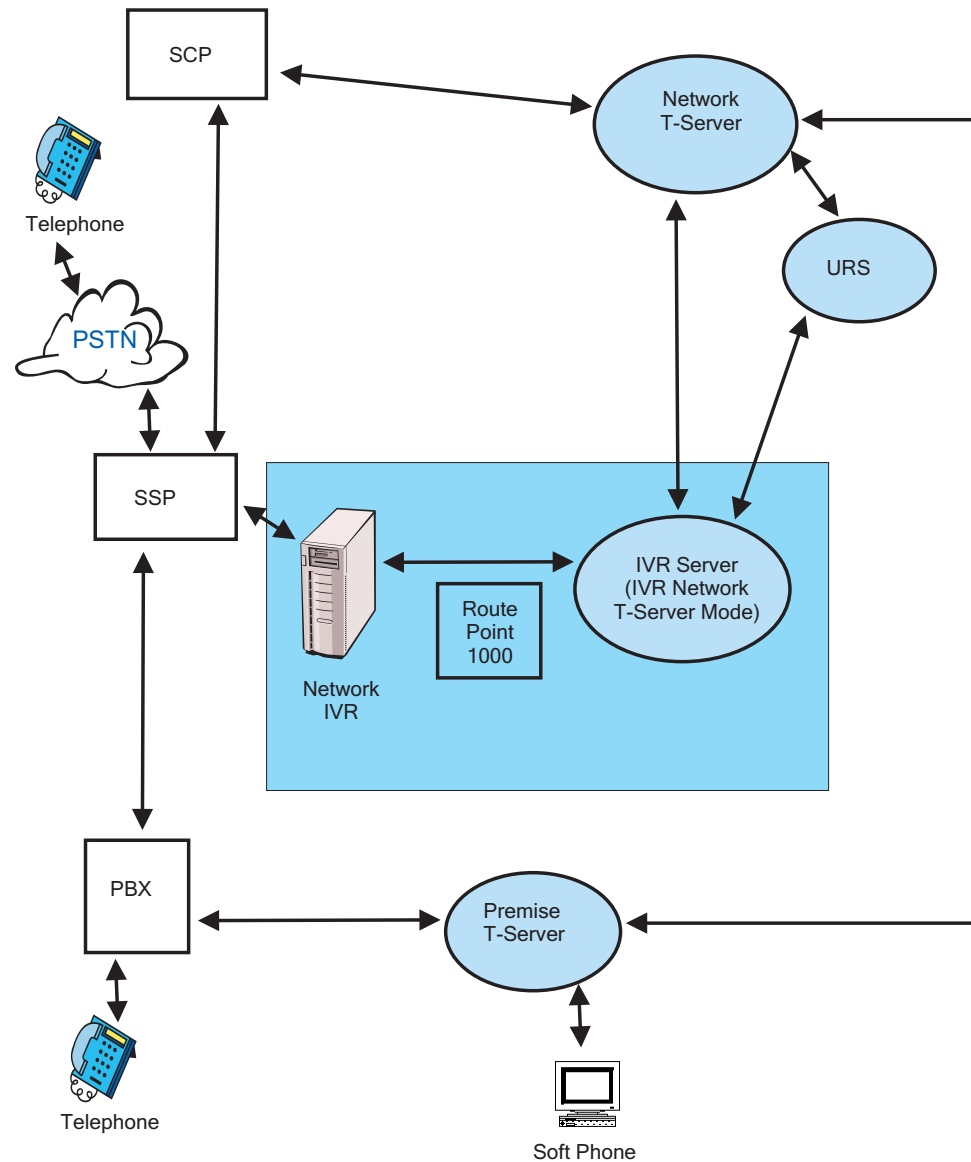


Figure 18: IVR Network T-Server Sample Configuration

This sample IVR Network T-Server configuration uses Routing Point 1000, statistics, a Network T-Server, and routing services (through Universal Routing Server [URS]).

The procedures in this section explain the following steps in detail:

1. Configure the switching office.
2. Configure the switch.
 - a. Configure access codes.
 - b. Configure the DNs.

3. Configure the TServer_IVR_Network application.
4. Connect the Network T-Server and Stat Server applications to the TServer_IVR_Network application..

Note: A connection from the URS application to the TServer_IVR_Network application is necessary if you want to create routing strategies.

Configuring the Switching Office

To configure the IVR Network T-Server switching office in Configuration Manager:

1. Expand Environment, right-click Switching Offices, and select New > Switching Office. The New Switching Office Properties dialog box appears.
2. In the Name box, enter IVR Network T-Server Switching Office.
3. In the Switch Type box, select GenSpec XML.
4. Click OK to complete the configuration of the IVR Network T-Server switching office.

Configuring the Switch

To configure the IVR Network T-Server switch in Configuration Manager:

1. Expand Resources (or Tenant Name), right-click Switches, and select New > Switch. The New Switch Properties dialog box appears.
2. Click the Folder icon next to the Switching Office box, select IVR Network Switching Office, and click OK.
3. Click the Folder icon next to the T-Server box, select TServer_IVR_Network_810, and click OK.
4. Click OK to complete the configuration of the IVR Network T-Server switch.

Configuring Access Codes

For information about how to configure access codes for the IVR Network T-Server switch, see “Switches and Access Codes” on [page 92](#).

Configuring DNs

For information about how to configure DNs for the IVR Network T-Server switch, see “Configuring the DNs” on [page 265](#).

Configuring the TServer_IVR_Network Application

To configure the TServer_IVR_Network application in Configuration Manager:

1. Expand Environment, right-click Applications, and select New > Application. The Browse dialog box appears.
2. Select TServer_IVR_Network_810, and then click OK. The New TServer_IVR_Network_810 Properties dialog box appears.
3. In the Name box, enter TServer_IVR_Network.
4. Click Apply to save your changes on this tab.
5. Click the Switches tab.
6. Select IVR Network Switch.
7. Click Apply to save your changes on this tab.
8. Click the Server Info tab.
9. In the Working Directory and Command Line boxes, enter the applicable information, as described in [Steps 17–19 on page 160](#).
10. Click Apply to save your changes on this tab.
11. Click the Server Info tab.
12. In the Host box, enter the host that the TServer_IVR_Network application will use.
13. In the Ports box, add a new port number that the TServer_IVR_Network application will use.
14. Click Apply to save your changes on this tab.
15. Click the Options tab.
16. Create the following new sections (if they have not already been defined):

• CallIdSap	• IServerGLMSap
• extrouter	• License
• gli	• Log
• gli_server	• Timers
• gli_server_group_1	• TServer
• IServer	• TServerClientSap
17. Add configuration options for each of these sections, as described in Chapter 12 on [page 231](#).

Note: You must set the operation-mode option in the IServerGLMSap section to NTS.

18. Click Apply to save your changes on this tab.
19. Click OK to complete the configuration of the TServer_IVR_Network application.

Connecting the TServer_IVR_Network Application to Other Applications

To connect the TServer_IVR_Network application to other applications in the Genesys network in Configuration Manager:

1. Expand Environment > Applications, right-click TServer_IVR_Network, and select Properties. The TServer_IVR_Network_810 Properties dialog box appears.
2. Click the Connections tab.
3. Click Add to add connections (if required) to the following:
 - Network T-Server application
 - Premise T-Server application
 - Message Server
4. Click OK to save your connections.

Network Mode—Logs

When operating in Network mode, you can notice these differences in the logs:

- You will see an I-Server application not found message that can be ignored.
- You can note that the parameter operation-mode is set to NTS in the log.
- No “PortMap” is created or shown.



Appendix

B

GLI Layer Configuration

This appendix provides GDI Link Interface (GLI) Layer configuration information. It contains the following sections:

- [Introduction, page 281](#)
- [Concepts, page 281](#)
- [Configuration, page 284](#)

Introduction

GLI is a subset of the Intelligence Service Control Point (ISCP) Generic Data Interface Specifications for TCP/IP, as defined in Chapter 2 of Telcordia's Special Report SR-3389. The full title of the document is *ISCP Generic Data Interface Specification for TCP/IP: Using GR-246-CORE ANSI TCAP Encoding*. You can obtain it directly from the Telcordia Store, at www.telcordia.com.

Concepts

This section discusses the following concepts as they relate to GLI layer configuration:

- TCP/IP connection
- GLI protocol
- Link/circuit behavior
- Circuit groups
- Circuit failover
- Security

TCP/IP Connection

The GLI layer provides a mechanism for forming TCP/IP connections between clients and servers through UNIX sockets. Essentially, a *socket* is a numbered port that is opened on a network interface for a particular application. The port number can be set either dynamically in a configuration file, or statically.

Sockets can be opened in one of two modes: passive (server) or active (client).

Servers are responsible for providing one or more services to clients; however, clients must use the correct protocol to communicate to the server. In this way, after a server socket is opened, the server will listen for client requests on its port.

Clients that want to communicate with a server must open a connection to the correct server socket. This is done by opening an active socket (usually, any client port will suffice) and connecting it to the server socket. At this point, the server accepts the connection, and it can optionally check security constraints. If the check fails, the server can take appropriate action, including closing the connection. Otherwise, the client can now use the connection to make requests to the server. If a connection fails, it is the client's responsibility to reconnect to the server.

Note: In the scenario shown in [Figure 19](#), IVR Server is the server, and IVR Driver is the client.

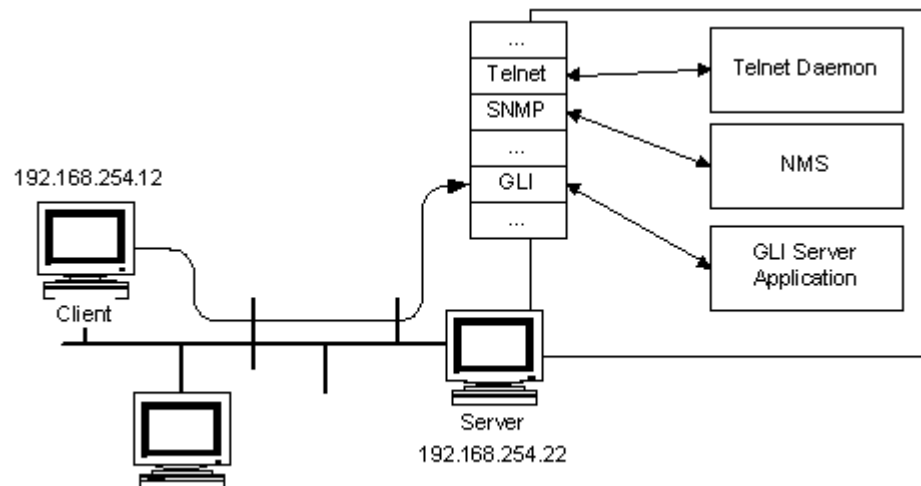


Figure 19: TCP/IP Connection

GLI Protocol

GLI uses a six-byte header on messages between the client and the server. This header identifies four different message types: `KeepAliveRequest`, `KeepAliveResponse`, `Error`, and `Data`. The next section describes these message types in more detail.

Link/Circuit Behavior

A GLI connection is also known as a link or circuit. After a GLI circuit is opened, the GLI client can send Data messages to the server. These GLI Data messages are used to encapsulate request and reply messages that were encoded using some other protocol.

The server sends GLI Error messages to the client to indicate certain severe error conditions. The client is responsible for closing the circuit.

Because TCP/IP timeouts are an unreliable mechanism for recognizing the failure of a circuit (for example, a half-open connection), KeepAlive messages (also known as heartbeat messages) can be used to determine the health of the circuit. Either side of a circuit can send a KeepAliveRequest message. The opposite side is responsible for replying with a KeepAliveResponse message. If a response is not received within a configured time period, and after a specified number of attempts, the requesting side can close the circuit.

Circuit Groups

Multiple circuits can be established between a client and server. The GLI Layer enables these circuits to be organized into logical groups. When multiple circuits are present in a group, Load Balancing on initial requests can be performed in a continuously repeating fashion. Replies are sent on the same circuit, unless a link failure occurs.

Circuit groups are particularly useful on servers that have multiple network interfaces. A server with link failover enabled can group circuits across different network interfaces. This provides redundant network links between the client and server (see Group 1 in [Figure 20](#)).

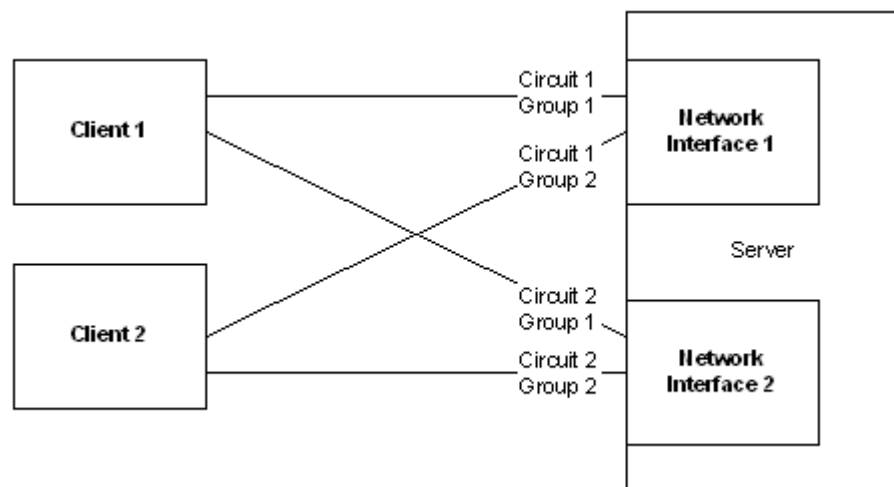


Figure 20: Circuit Groups

Circuit Failover

Circuit failover is a special flag for circuit groups that indicates whether the circuits in a group can be treated as equivalent in the event of a circuit failover. In other words, if a circuit fails while a server is replying to a client, the failover flag tells the server whether another circuit in the group can be used to send the reply.

Circuit failover requires that transaction information be shared across all the clients that are attached to the circuits in a circuit group.

Security

Security on the server side is based on a list of clients that are allowed to connect to the server. Specific IP addresses and port numbers can be included.

Configuration

Table 18 provides the GLI Layer configuration options, with values, defaults, and descriptions.

Table 18: GLI Layer Configuration Options

Option Name	Valid Values	Default Value	Description
Section: [gli]			
gli-mode	server, client	None	Specifies the type of socket to open.
gli-keep-alive-interval	1.. <i>n</i>	-1	Specifies the amount of time (in seconds) to wait before sending <code>KeepAliveRequests</code> . A value of -1 disables this feature.
gli-keep-alive-tries	1.. <i>n</i>	1	Specifies the number of times to retry sending <code>KeepAliveRequests</code> before considering the circuit as failed.
gli-reconnect-delay	1.. <i>n</i>	5	Specifies the amount of time (in seconds) to wait before attempting to reopen a failed socket.
gli-flow-control-bound			Reserved for future use.
gli-queue-max-timeout			Reserved for future use.

Table 18: GLI Layer Configuration Options (Continued)

Option Name	Valid Values	Default Value	Description
Server Options			
Section: [gli_server]			
gli-server-mode	circuit	None	Specifies that the server can open multiple circuits.
gli-n-servers	1...8	None	Specifies the number of circuit groups that the server can open. There will be one configured section for each server group.
Section: [gli_server_group_1-8]			
gli-server-address	host:port	None	Specifies, using a comma-separated list, the sockets that the server can use for this group. Any connection that is made to this socket forms a circuit that is part of this group. You can use either an IP address or a host name as an address, and you must supply a port number. To use the default interface, specify 0.0.0.0 as the IP address.
gli-client-list	host:port	None	Specifies, using a comma-separated list, the client sockets that are allowed to connect to the server. You can use either an IP address or a host name as an address, and you can use an asterisk (*) to indicate any port number. If this option is not supplied, or if it is empty, any client can connect to the server.
gli-circuit-failover	on off	on	<p>Specifies whether link failover is allowed for circuits in this group.</p> <p>When set to on, any attempt to send a message on a failed circuit causes the message to be sent on another active circuit in the group.</p> <p>When set to off, any attempt to send a message on a failed circuit causes the message not to be delivered.</p> <p>If all the applications that connect to this group can process messages from any circuit, you should set this option to on.</p> <p>If all the applications that connect to the group are independent of one another, you should set this option to off.</p>

Table 18: GLI Layer Configuration Options (Continued)

Option Name	Valid Values	Default Value	Description
gli-tls-cert	None	N/A	For Windows, contains the thumbprint that is obtained from a user certificate generated for the host. For UNIX, contains the path and file name to a .pem encoded file that contains the host certificate.
gli-tls-cert-key	None	N/A	For UNIX only, contains the path and file name to a .pem encoded file that contains the host private key.
gli-tls-trusted-ca	None	N/A	For UNIX only, contains the path and file name to a .pem encoded file that contains the Certificate Authority (CA) certificate.



Appendix

C

Configuring Application Connections

This appendix describes the connections that you must configure between applications for each of the three IVR Server configuration modes (IVR-In-Front, IVR-Behind-Switch, and IVR Network T-Server). You configure these connections in Configuration Manager, on the **Connections** tab of the **Properties** dialog box for each **Application** object.

This appendix contains the following sections:

- [IVR-In-Front Connections, page 287](#)
- [IVR-Behind-Switch Connections, page 288](#)
- [IVR Network T-Server Connections, page 289](#)

IVR-In-Front Connections

[Table 19](#) shows the application connections that are required for the IVR-In-Front configuration mode.

Note: The IVR-In-Front configuration does not include a premise T-Server.

Table 19: IVR-In-Front Application Connections

Add a Connection to This Application						
On This Application's Connections Tab	TServer_IVR	I-Server	Premise T-Server	URS	Stat Server	Message Server
TServer_IVR						X
I-Server	X				X	
IVR_Driver						X
URS	X				X	X
Stat Server	X					X

IVR-Behind-Switch Connections

Table 20 shows the application connections that are required for the IVR-Behind-Switch configuration mode.

Note: If you are using a dual configuration mode (IVR-Behind-Switch and IVR-In-Front), you must also create a connection from the I-Server application to the TServer_IVR application.

Table 20: IVR-Behind-Switch Application Connections

Add a Connection to This Application						
On This Application's Connections Tab	TServer_IVR	I-Server	Premise T-Server	URS	Stat Server	Message Server
TServer_IVR						X
I-Server	X		X		X	X
IVR_Driver						X
Premise T-Server						X
URS			X		X	X
Stat Server			X			X

IVR Network T-Server Connections

Table 21 shows the application connections that are required for the IVR Network T-Server configuration mode.

Note: The IVR Network T-Server configuration uses a Network T-Server instead of a Premise T-Server. It also uses the TServer_IVR_Network application instead of the TServer_IVR application. This configuration does not include an I-Server application or an IVR_Driver application.

Table 21: IVR Network T-Server Application Connections

Add a Connection to This Application						
On This Application's Connections Tab	TServer_IVR_Network	I-Server	Network T-Server	URS	Stat Server	Message Server
TServer_IVR_Network			X			X
URS	X		X		X	X
Stat Server	X		X			X
Network T-Server	X					



Supplements

Related Documentation Resources

The following resources provide additional information that is relevant to this software. Consult these additional resources as necessary.

IVR Drivers

- A series of *IVR Driver Systems Administrator's Guide*, which provide information about how to install and configure specific IVR drivers supplied by third-party vendors.
- Release Notes and Product Advisories for this product, which are available on the Genesys Technical Support website at <http://genesyslab.com/support>

Genesys

- *Genesys Technical Publications Glossary*, which ships on the Genesys Documentation Library DVD and which provides a comprehensive list of the Genesys and computer-telephony integration (CTI) terminology and acronyms used in this document.
- *Genesys Migration Guide*, which ships on the Genesys Documentation Library DVD, and which provides documented migration strategies for Genesys product releases. Contact Genesys Technical Support for more information.

Information about supported hardware and third-party software is available on the Genesys Technical Support website in the following documents:

- [*Genesys Supported Operating Environment Reference Manual*](#)
- [*Genesys Supported Media Interfaces Reference Manual*](#)

Consult these additional resources as necessary:

- *Genesys Hardware Sizing Guide*, which provides information about Genesys hardware sizing guidelines for the Genesys 8.x releases.

- *Genesys Interoperability Guide*, which provides information on the compatibility of Genesys products with various Configuration Layer Environments; Interoperability of Reporting Templates and Solutions; and Gplus Adapters Interoperability.
- *Genesys Licensing Guide*, which introduces you to the concepts, terminology, and procedures relevant to the Genesys licensing system.
- *Genesys Database Sizing Estimator 8.x Worksheets*, which provides a range of expected database sizes for various Genesys products.
- *IVR SDK 8.1 C Developer's Guide*
- *IVR SDK 8.1 XML Developer's Guide*
- *Voice Platform SDK 8.x .NET (or Java) API Reference*
- *Framework 8.1 Deployment Guide*
- *Framework 8.1 T-Server Deployment Guide*
- *Framework 8.1 Configuration Options Reference Manual*

For additional system-wide planning tools and information, see the release-specific listings of System Level Documents on the Genesys Technical Support website, accessible from the [system level documents by release](#) tab in the Knowledge Base Browse Documents Section.

Genesys product documentation is available on the:

- Genesys Technical Support website at <http://genesyslab.com/support>.
- Genesys Documentation wiki at <http://docs.genesyslab.com/>.
- Genesys Documentation Library DVD and/or the Developer Documentation CD, which you can order by e-mail from Genesys Order Management at orderman@genesyslab.com

Document Conventions

This document uses certain stylistic and typographical conventions—introduced here—that serve as shorthands for particular kinds of information.

Document Version Number

A version number appears at the bottom of the inside front cover of this document. Version numbers change as new information is added to this document. Here is a sample version number:

80fr_ref_06-2008_v8.0.001.00

You will need this number when you are talking with Genesys Technical Support about this product.

Screen Captures Used in This Document

Screen captures from the product graphical user interface (GUI), as used in this document, may sometimes contain minor spelling, capitalization, or grammatical errors. The text accompanying and explaining the screen captures corrects such errors *except* when such a correction would prevent you from installing, configuring, or successfully using the product. For example, if the name of an option contains a usage error, the name would be presented exactly as it appears in the product GUI; the error would not be corrected in any accompanying text.

Type Styles

[Table 22](#) describes and illustrates the type conventions that are used in this document.

Table 22: Type Styles

Type Style	Used For	Examples
Italic	<ul style="list-style-type: none"> Document titles Emphasis Definitions of (or first references to) unfamiliar terms Mathematical variables <p>Also used to indicate placeholder text within code samples or commands, in the special case where angle brackets are a required part of the syntax (see the note about angle brackets on page 294).</p>	<p>Please consult the <i>Genesys Migration Guide</i> for more information.</p> <p>Do <i>not</i> use this value for this option.</p> <p>A <i>customary and usual</i> practice is one that is widely accepted and used within a particular industry or profession.</p> <p>The formula, $x + 1 = 7$ where x stands for . . .</p>
Monospace font (Looks like teletype or typewriter text)	<p>All programming identifiers and GUI elements. This convention includes:</p> <ul style="list-style-type: none"> The <i>names</i> of directories, files, folders, configuration objects, paths, scripts, dialog boxes, options, fields, text and list boxes, operational modes, all buttons (including radio buttons), check boxes, commands, tabs, CTI events, and error messages. The values of options. Logical arguments and command syntax. Code samples. <p>Also used for any text that users must manually enter during a configuration or installation procedure, or on a command line.</p>	<p>Select the Show variables on screen check box.</p> <p>In the Operand text box, enter your formula.</p> <p>Click OK to exit the Properties dialog box.</p> <p>T-Server distributes the error messages in EventError events.</p> <p>If you select true for the inbound-bsns-calls option, all established inbound calls on a local agent are considered business calls.</p> <p>Enter exit on the command line.</p>
Square brackets ([])	A particular parameter or value that is optional within a logical argument, a command, or some programming syntax. That is, the presence of the parameter or value is not required to resolve the argument, command, or block of code. The user decides whether to include this optional information.	<code>smcp_server -host [/flags]</code>
Angle brackets (< >)	<p>A placeholder for a value that the user must specify. This might be a DN or a port number specific to your enterprise.</p> <p>Note: In some cases, angle brackets are required characters in code syntax (for example, in XML schemas). In these cases, italic text is used for placeholder values.</p>	<code>smcp_server -host <confighost></code>



Index

Symbols

[] (square brackets)	294
< > (angle brackets)	294
<key name>	
common log option	226

A

Access Code	
configuration	94
defined	92
access codes	143
ACD queues, defining	272
active-release	
configuration option	236
adding	
access codes	143
I-Server connections	113, 269, 275
IVR ports	136
IVR_Driver connections	109
server applications	163
TServer_IVR_Network connections	279
addp-remote-timeout	
configuration option	202
addp-timeout	
configuration option	203
addp-trace	
configuration option	203
Advanced Disconnect Detection Protocol	17, 39
agent	
logout	28
state management	28
state status management	29
agent reservation	
defined	44
AgentControl section	255
LegacyMode option	255
AgentLogin section	244
IgnoreReady option	244
ReadyWorkMode option	244

AgentLogout section	244
LogoutOnDisable option	245
LogoutOnShutdown option	244
TimePerLogout option	245
agent-reservation section	
configuration options	190–191
alarm	
common log option	217
all	
common log option	216
angle brackets	294
ANI	57
ani-distribution	
configuration option	182
Annex tab	142, 255, 259
API Reference Tree	30
Application objects	
multi-site operation	91
application templates	
I-Server requirements	106
I-Server, importing	110
IVR_Driver requirements	106
IVR_Driver, importing	107
overview	106
TServer_IVR requirements	106
TServer_IVR_Network requirements	106
TServer_IVR_Network, importing	119
TServer_IVR, importing	114
applications	
enabling	136
I-Server options	244
I-Server, configuring	111, 151, 267, 273
I-Server, defining	111
IVR_Driver options	248
IVR_Driver security	158
IVR_Driver, configuring	108, 156
IVR_Driver, defining	107
names	178, 179
TServer_IVR security	155
TServer_IVR_Network security	161

TServer_IVR_Network, configuring . 121, 158, 278
 TServer_IVR_Network, defining 120
 TServer_IVR, configuring . .115, 152, 268, 274
 TServer_IVR, defining 114
 app-name
 configuration option 236, 245
 architecture 19
 Aspect driver 266
 assigning
 I-Server hosts 112
 IVR_Driver hosts 108
 network switch 122
 TServer_IVR hosts 117
 TServer_IVR_Network hosts 122
 virtual switch 116
 associations. See connections
 audience, for document 14
 Auto-Login feature 150
 AutoLogin section 259
 AgentId option 259
 Password option 260
 Queue option 259
 SetLoggedIn option 260
 SetReady option 260
 availability of services 27

B

background-processing
 configuration option 182
 background-timeout
 configuration option 183
 backup servers 165
 backup-sync section
 configuration options 202–203
 brackets
 angle 294
 square 294
 buffering
 common log option 210

C

call flow 19
 call-cleanup section
 configuration options 204–205
 called-num-subset
 configuration option 236
 CallIdSap section 233
 input-network-call-id-key 233
 callinfo_all_returns_3rdpartydn
 configuration option 248
 callinfo_all_returns_uuid
 configuration option 249

calls, monitoring 22
 call-timer-timeout
 configuration option 242
 cast-type
 configuration option 56, 193
 CDN 63
 centralized logging 118, 124
 changes from 8.0 to 8.1
 common configuration options 229
 IVR Server configuration options 260
 T-Server common configuration options 207
 checkout-interval
 configuration option 238
 check-point
 common log option 214
 check-tenant-profile
 configuration option 183
 circuits 283, 284
 cleanup-idle-tout
 configuration option 204
 Code property 94, 95
 cof-ci-defer-create
 configuration option 198
 cof-ci-defer-delete
 configuration option 198
 cof-ci-req-tout
 configuration option 72, 198
 cof-ci-wait-all
 configuration option 199
 cof-feature
 configuration option 199
 cof-rci-tout
 configuration option 199
 collect-lower-priority-requests
 configuration option 190
 command line parameters
 nco X/Y 178
 commenting on this document 14
 common configuration options 210–230
 changes from 8.0 to 8.1 229
 common section 229
 disable-rbac 227
 enable-async-dns 229
 hangup-restart 228
 heartbeat-period 227
 heartbeat-period-thread-class-<n> 228
 log section 210–224
 log-extended section 224–226
 log-filter section 226
 log-filter-data section 226–227
 mandatory 210
 rebind-delay 229
 security section 227
 setting 209
 sml section 227–229
 suspending-wait-timeout 228

- common log options 210–226
 - <key name> 226
 - alarm 217
 - all 216
 - buffering 210
 - check-point 214
 - compatible-output-priority 215
 - debug 219
 - default-filter-type 226
 - expire 211
 - interaction 218
 - keep-startup-file 212
 - level-reassign-<eventID> 224
 - level-reassign-disable 226
 - log section 210–224
 - log-extended section 224–226
 - log-filter section 226
 - log-filter-data section 226–227
 - mandatory options 210
 - memory 214
 - memory-storage-size 215
 - message_format 212
 - messagefile 212
 - print-attributes 214
 - segment 211
 - setting 209
 - spool 215
 - standard 218
 - time_convert 213
 - time_format 213
 - trace 218
 - verbose 210
 - x-conn-debug-all 224
 - x-conn-debug-api 223
 - x-conn-debug-dns 223
 - x-conn-debug-open 222
 - x-conn-debug-security 223
 - x-conn-debug-select 222
 - x-conn-debug-timers 222
 - x-conn-debug-write 222
- common options
 - common log options 210–226
 - common section 229
 - mandatory options 210
 - sml section 227–229
- common section
 - common options 229
- compat65
 - configuration option 249
- compatibility 30
- compatible-output-priority
 - common log option 215
- components 30, 177
- compound-dn-representation
 - configuration option 200
- Configuration Layer 17, 30
- configuration modes 106
- examples 263
 - IVR Network T-Server 106, 276
 - IVR-Behind-Switch 19, 20, 106, 270
 - IVR-In-Front 19, 20, 27, 106, 263
 - Network T-Server 19, 21
- configuration options
 - addp-remote-timeout 202
 - addp-timeout 203
 - addp-trace 203
 - agent-reservation section 190–191
 - ani-distribution 182
 - background-processing 182
 - background-timeout 183
 - backup-sync section 202–203
 - call-cleanup section 204–205
 - cast-type 193
 - changes from 8.0 to 8.1 207, 260
 - check-tenant-profile 183
 - cleanup-idle-tout 204
 - cof-ci-defer-create 198
 - cof-ci-defer-delete 198
 - cof-ci-req-tout 198
 - cof-ci-wait-all 199
 - cof-feature 199
 - cof-rci-tout 199
 - collect-lower-priority-requests 190
 - common log options 210–226
 - common options 210–230
 - compound-dn-representation 200
 - consult-user-data 183
 - customer-id 184
 - default-dn 194
 - default-network-call-id-matching 200
 - direct-digits-key 194
 - dn-for-unexpected-calls 195
 - dn-scope 86, 184
 - epp-tout 87, 201
 - event-propagation 201
 - extrouter section 191–202
 - handle-vsp 202
 - inbound-translator-<n> 201
 - license section 187–190
 - local-node-id 199
 - log-trace-flags 185
 - management-port 185
 - mandatory options 210
 - match-call-once 192
 - merged-user-data 185
 - network-request-timeout 195
 - notify-idle-tout 204
 - num-of-licenses 187
 - num-sdn-licenses 188
 - periodic-check-tout 204
 - propagated-call-type 86, 186
 - protocol 203

- reconnect-tout 192
- register-attempts 195
- register-tout 195
- reject-subsequent-request 191
- report-connid-changes 192
- request-collection-time 191
- request-tout 195
- reservation-time 191
- resource-allocation-mode 196
- resource-load-maximum 196
- route-dn 196
- rule-<n> 205
- security section 206
- server-id 186
- setting 181
 - common 209
- sync-reconnect-tout 203
- tcs-queue 197
- tcs-use 198
- timeout 197
- timeout value format 206
- Translation Rules section 205
- TServer section 182–187
- use-data-from 193
- use-implicit-access-numbers 197
- user-data-limit 187
- configuring
 - Auto-Login feature 150
 - DNS 145, 265
 - GLI layer 281
 - I-Server application 111, 151, 267, 273
 - IVR devices 146, 266, 271
 - IVR Drivers 177
 - IVR manually 135
 - IVR port exceptions 137
 - IVR ports 147, 266, 272
 - IVR_Driver application 108, 156
 - multiple IVR ports 148
 - multi-site operation 91–104
 - steps 91
 - options 231
 - switches 143, 265, 271, 277
 - switching offices 142, 265, 271, 277
 - tasks 138
 - TServer_IVR application 115, 152, 268, 274
 - TServer_IVR_Network application 121, 158, 278
 - wizard 131
- connecting
 - I-Server application with IVR 269, 275
- connections
 - I-Server, adding 113, 269, 275
 - IVR_Driver, adding 109
 - TServer_IVR_Network, adding 279
- consult-user-data
 - configuration option 183

- conventions
 - in document 293
 - type styles 294
- CONVERSANT driver 106, 266
- creating
 - IVR port range 133
 - network switch 122
 - network switching office 121
 - single IVR port 147
 - virtual switch 116
 - virtual switching office 115
- customer-id
 - configuration option 184

D

- Database Access Point 119, 124
- databases 31
- DataTransport section 255
 - load_sharing_1servers option 258
 - log_file_backup_amount 256
 - log_file_name option 256
 - log_file_size option 256
 - log_print_date option 257
 - log_print_driver_selector option 258
 - log_print_hb option 258
 - log_print_level option 257
 - log_print_login_requests option 258
 - log_print_name option 256
 - log_print_recv option 258
 - log_print_send option 258
 - log_print_time option 257
 - log_print_time_ms option 257
 - log_print_timeouts option 256
 - log_print_udata option 257
 - socket_activity_timer option 259
 - time_recon_is option 256
- debug
 - common log option 219
 - configuration option
 - pgf-debug section 239
- Default Access Code
 - configuration 93
 - defined 92
- default access codes 143
- default-dn
 - configuration option 194
- default-filter-type
 - common log option 226
- default-network-call-id-matching
 - configuration option 200
- defer-tlib-events
 - configuration option 242
- defining
 - ACD queues 272

- I-Server application 111
- I-Server start parameters 112
- IVR hosts 108, 112, 117, 123, 151, 153, 156, 159
- IVR_Driver application. 107
- IVR_Driver start parameters. 109
- routing points 273
- TServer_IVR application 114
- TServer_IVR start parameters 118
- TServer_IVR_Network application 120
- TServer_IVR_Network start parameters 123
- deployment. 30
- destination location. 50
- destination T-Server 56
- direct-ani
 - ISCC transaction type 57, 65
- direct-callid
 - ISCC transaction type 58, 65
- direct-digits
 - transaction type 65
- direct-digits-key
 - configuration option 194
- direct-network-callid
 - ISCC transaction type 58, 65
- direct-notoken
 - ISCC transaction type 59, 65
- direct-uui
 - ISCC transaction type 59, 65
- disable-rbac
 - common configuration option 227
- dn-for-unexpected-calls
 - configuration option 195
- dnis-pool
 - in load-balancing mode 61
 - ISCC transaction type 52, 60, 65
- DNs
 - availability 27
 - configuring. 145, 265
 - configuring for multi-sites 98
- dn-scope
 - configuration option 86, 184
- document
 - audience. 14
 - change history. 15
 - conventions 293
 - errors, commenting on 14
 - version number 293
- dtd-file
 - configuration option 236

E

- editing access codes 143
- enable-async-dns
 - common configuration option 229
- enabling
 - Annex tab 142

- IVR applications. 136
- network logging 118, 124
- epp-tout
 - configuration option 87, 201
- Event Propagation
 - defined 83
- EventAttachedDataChanged 84
- event-propagation
 - configuration option 201
- example configurations 263
- exceptions, IVR port configuration. 137
- expire
 - common log option 211
- extrouter section 232
 - configuration options 191–202
 - configuring for multi-site operation 92
 - configuring party events propagation 88
 - configuring the Number Translation feature 81

F

- failover 284
- figures
 - hot standby redundancy 168
 - Multiple-to-Point mode 64
 - Point-to-Point mode. 63
 - steps in ISCC/Call Overflow 71
- flow control
 - configuration option 30
- flow-control
 - configuration option 236
- font styles
 - italic 294
 - monospace 294

G

- GDI Link Interface. *See* GLI
- Genesys
 - environment. 30
 - licensing. 31, 136
 - migration 31
 - Wizard Manager 131
- getreply_with_location
 - configuration option 249
- GLI
 - layer configuration 281
 - protocol 282
 - security 284
- gli section 233
 - gli-keep-alive-interval option 233
 - gli-keep-alive-tries option 233
 - gli-mode option 233
 - gli-reconnect-delay option 234
- gli_server section 234

gli-n-servers options	234
gli-server-mode option	234
gli_server_group_<n> section	234
gli-circuit-failover option	234
gli-client-list option	234
gli-server-address option	235
gli-tls-cert option	235

H

HA

See also high availability

See hot standby

handle-vsp	
configuration option	202
hangup-restart	
common configuration option	228
hardware prerequisites	31
heartbeat-period	
common configuration option	227
heartbeat-period-thread-class-<n>	
common configuration option	228
hide-user-data	
configuration option	239
hosts	
defining 108, 112, 117, 123, 151, 153, 156, 159	
I-Server, assigning	112
IVR_Driver, assigning	108
names	178
TServer_IVR_Network, assigning	122
TServer_IVR, assigning	117
hot standby	40, 117, 123, 165
defined	40
figure	168
T-Server configuration	173

I

importing templates	
I-Server application	110
IVR_Driver application	107
TServer_IVR application	114
TServer_IVR_Network application	119
inbound-translator-<n>	
configuration option	201
InFront section	245
UseQueue option	245
input-network-call-id-key	
configuration option	233
installing	
IVR Drivers	177
IVR Server	127
prerequisites	31
setup tasks	105
UNIX process	127

Windows process	128
intended audience	14
Inter Server Call Control	50–69
Inter Server Call Control/Call Overflow	69–73
interaction	
common log option	218
interoperability	30
ISCC	
destination T-Server	56
origination T-Server	56
ISCC transaction types	51, 56
direct-ani	57, 65
direct-callid	58, 65
direct-digits	65
direct-network-callid	58, 65
direct-notoken	59, 65
direct-uuui	59, 65
dnis-pool	60, 65
in load-balancing mode	61
pullback	61, 65
reroute	62, 65
route	63, 65
route-uuui	64
supported	65
ISCC/COF	
supported	70
iscc-xaction-type	51
I-Server application	
adding connections	113, 269, 275
assigning hosts	112
configuring	111, 151, 267, 273
connecting IVR devices	269, 275
defining	111
defining start parameters	112
importing template	110
options	244
template requirements	106
Iserver section	
active-release option	236
app-name option	236
called-num-subset option	236
dtd-file option	236
flow-control option	236
peer-list option	237
peer-mode-dn option	237
peer-mode-timer	237
report-dn-status option	237
route-in-place	
configuration option	237
transport-port option	238
iserver_mode_hotstandby	
configuration option	250
IServerGLMSap section	238
checkout-interval option	238
operation-mode option	238
italics	294

IVR

- API Reference Tree 30
- components 177
- configuring devices 146, 266, 271
- configuring drivers 177
- installing drivers 177
- I-Server, connecting devices 269, 275
- overview 13, 17
- ports. *See* ports
- SDK 23, 30
- wizard 105
- IVR annex tab
 - DriverIgnoreReady option 254
 - DriverReadyWorkMode option 254
 - DriverRetryTimeout option. 254
- IVR drivers
 - CONVERSANT 106
 - defined. 23
 - deploying 30
- IVR In-Front
 - DN availability 27
- IVR Library 23
- IVR Network T-Server
 - configuration mode 21, 106, 276
 - overview 19
- IVR Server 13
 - compatibility 30
 - configuration modes 19
 - defined. 19
 - deploying 30
 - interoperability 30
 - link status 27
 - starting 178
 - stopping 179
- IVR_Driver application
 - adding connections 109
 - assigning hosts 108
 - configuring 108, 156
 - defining 107
 - defining start parameters 109
 - importing template. 107
 - options. 248
 - security 158
 - template requirements. 106
- ivr_server_interface section 248
 - callinfo_all_returns_3rdpartydn option 248
 - callinfo_all_returns_uuid option 249
 - compat65 option. 249
 - getreply_with_location option 249
 - iserver_mode_hotstandby option 250
 - load_sharing_iservers option 250
 - load_sharing_iservers_client_hosts option 250
 - load_sharing_iservers_client_ports option 250
 - socket_activity_timer option. 250
 - time_recon_is option 251

IVR-Behind-Switch

- configuration mode 20, 106, 270
- overview 19

IVR-In-Front

- configuration mode 20, 106, 263
- overview 19

K

keep-startup-file

- common log option 212

L

level-reassign-<eventID>

- common log option 224

level-reassign-disable

- common log option 226

license section

- configuration options 187–190

licensing

- 31, 136

link status

- 27

links

- 283

load balancing

- 23, 28

load_sharing_iservers

- configuration option. 250

load_sharing_iservers_client_hosts

- configuration option. 250

load_sharing_iservers_client_ports

- configuration option. 250

LoadBalance section

- app-name option 245

local-node-id

- configuration option 199

location parameter

- 50

log configuration options

- 210–216

Log DB Server

- 119, 124

log section

- 232

common log options

- 210–224

log_content section

- 251

log_print_agent_login option

- 251

log_print_date option

- 251

log_print_driver_selector option

- 251

log_print_hb option

- 252

log_print_level option

- 252

log_print_login_requests option

- 252

log_print_name option

- 252

log_print_recv option

- 252

log_print_send option

- 253

log_print_time option

- 253

log_print_time_ms option

- 253

log_print_timeouts option

- 253

log_print_udata option

- 253

log-extended section

- common log options 224–226

log-filter section
 common log options 226
 log-filter-data section
 common log options 226–227
 logging, network 118, 124
 log-trace-flags
 configuration option 185

M

Management Layer 17, 30
 management-port
 configuration option 185
 managing
 agent state 28
 agent state status 29
 service availability 27
 services 179
 mandatory options
 configuration options 182
 manual configuration 135
 match-call-once
 configuration option 192
 Media Layer 17, 30
 memory
 common log option 214
 memory-storage-size
 common log option 215
 merged-user-data
 configuration option 185
 Message Server 119, 124
 message_format
 common log option 212
 messagefile
 common log option 212
 migration 31, 132
 monitoring calls 22
 monospace font 294
 Multiple-to-One mode 63
 Multiple-to-Point mode 63, 64

N

NAT/C feature 81
 nco X/Y
 command line parameter 178
 network
 call monitoring 22
 logging 118, 124
 switch, assigning 122
 switch, creating 122
 switching offices 121
 network attended transfer/conference 81
 network-request-timeout
 configuration option 195

new in this release 18
 notify-idle-tout
 configuration option 204
 Number Translation feature 73–81
 number translation rules 74
 num-of-licenses
 configuration option 187
 num-sdn-licenses
 configuration option 188

O

One-to-One mode 63
 operating systems 31
 operation-mode
 configuration option 238
 options
 configuration 231
 I-Server application 244
 IVR_Driver application 248
 See also sections
 T-Server common 232
 origination location 50
 origination T-Server 56
 overview 13, 17

P

peer-list
 configuration option 237
 peer-mode-dn
 configuration option 237
 peer-mode-timer
 configuration option 237
 periodic-check-tout
 configuration option 204
 pgf section 239
 ptc-file option 239
 pgf-debug section 239
 debug option 239
 hide-user-data option 239
 place status algorithm 138
 Point-to-Point mode 63
 ports
 adding 136
 configuration exceptions 137
 configuring 147, 266, 272
 configuring multiple 148
 creating range 133
 creating single 147
 names 178
 voice treatment 137
 preconfiguration notes 136
 preface 13
 pre-installation tasks 105

prerequisites 31
 primary servers 165
 print-attributes
 common log option 214
 propagated-call-type
 configuration option 86, 186
 protocol
 configuration option 203
 ptc-file
 configuration option 239
 pullback
 ISCC transaction type 61, 65

R

rebind-delay
 common configuration option 229
 reconnect-tout
 configuration option 192
 redundancy
 hot standby 40, 165
 warm standby 40, 165
 redundancy types 173
 hot standby 40
 register-attempts
 configuration option 195
 register-tout
 configuration option 195
 reject-subsequent-request
 configuration option 191
 report-connid-changes
 configuration option 192
 report-dn-status
 configuration option 237
 report-dn-status option 27
 request-collection-time
 configuration option 191
 request-tout
 configuration option 52, 195
 reroute
 ISCC transaction type 62, 65
 reservation-time
 configuration option 191
 resource-allocation-mode
 configuration option 196
 resource-load-maximum
 configuration option 196
 route
 ISCC transaction type 52, 63, 65, 98
 route-dn
 configuration option 196
 route-in-place option 237
 route-uuui
 ISCC transaction type 64
 routing
 Inter Server Call Control 56–69

routing points 273
 rule-<n>
 configuration option 205

S

sections
 AgentControl 255
 AgentLogin 244
 AgentLogout 244
 AutoLogin 259
 CallIdSap 233
 DataTransport 255
 gli 233
 gli_server 234
 gli_server_group_<n> 234
 InFront 245
 IServerGLMSap 238
 ivr_server_interface 248
 LoadBalance 245
 log_content 251
 pgf 239
 pgf-debug 239
 Stat:<stat name> 246
 Timers 239
 TServerClientSap 242
 VirtualRoutePoints 247
 XmlSap 243
 security
 GLI 284
 IVR_Driver application 158
 TServer_IVR application 155
 TServer_IVR_Network application 161
 security section
 common configuration options 206, 227
 segment
 common log option 211
 server applications
 adding 163
 installing on UNIX 127
 installing on Windows 128
 server-id
 configuration option 186
 services
 availability 27
 managing 179
 setting configuration options
 common 209
 sml section
 common options 227–229
 socket_activity_timer
 configuration option 250
 sockets 282
 software prerequisites 31
 source-encoding
 configuration option 243

- spool
 - common log option 215
- square brackets 294
- standard
 - common log option 218
- start parameters
 - I-Server, defining 112
 - IVR_Driver, defining 109
 - TServer_IVR_Network, defining 123
 - TServer_IVR, defining 118
- starting IVR Server 178
- Stat Server place status algorithm 138
- Stat:<stat name> section
 - obj_id option 246
 - obj_type option 246
 - server_name option 246
 - stat_type option 246
 - time_profile option 246
 - update_frequency option 247
- status
 - agent state. 29
 - link. 27
- stopping IVR Server 179
- suspending-wait-timeout
 - common configuration option 228
- Switch objects
 - multi-site operation 91
- switch partitioning
 - defined. 86
 - T-Server support. 87
- switches
 - configuring. 143, 265, 271, 277
 - network, assigning. 122
 - network, creating 122
 - virtual, assigning. 116
 - virtual, creating 116
- Switching Office objects
 - multi-site operation 92, 93, 94, 98
- switching offices
 - configuring. 142, 265, 271, 277
 - network 121
 - virtual 115
- sync-reconnect-tout
 - configuration option 203

T

- Target ISCC
 - Access Code configuration 95
 - Default Access Code configuration 94
- target-encoding
 - configuration option 243
- tasks
 - configuration. 138
 - pre-installation. 105
- TCP/IP connections 282

- tcs-queue
 - configuration option 197
- tcs-use
 - configuration option 198
- templates. See application templates
- time_convert
 - common log option 213
- time_format
 - common log option 213
- time_recon_is
 - configuration option 251
- timeout
 - configuration option 52, 197
- timeout value format
 - configuration options 206
- Timers section 239
 - Call Timeout option 240
 - CME Update Timeout option 240
 - Registration Timeout option. 240
 - Retry Timeout option 240
 - Router Timeout option 240
 - Stat Timeout option 241
 - Stop Waiting Timeout option 241
 - Unregister Timeout option 241
 - valid time units 239
 - wait-for-ringing option 241
- TInitiateConference 50
- TInitiateTransfer 50
- TMakeCall 50
- TMuteTransfer 50
- trace
 - common log option 218
- transaction types (ISCC). 51, 56
 - supported 65
- transfer connect service 68
- Translation Rules section
 - configuration option 205
- transport-port
 - configuration option 238
- TRouteCall. 50
- trunk lines 63
- T-Server
 - common options 232
 - configuring Application objects
 - for multi-sites 91
 - configuring redundancy 170
 - HA. 173
 - high availability 173
 - hot standby 173
 - Library. 19
 - multi-site operation 91–104
 - redundancy 173
 - upgrading 136
 - warm standby 170
- TServer section 232
 - configuration options 182–187

TServer_IVR application 19
 assigning hosts 117
 configuring 115, 152, 268, 274
 defining 114
 defining start parameters 118
 importing template 114
 security 155
 template requirements 106
 TServer_IVR_Network application
 adding connections 279
 assigning hosts 122
 configuring 121, 158, 278
 defining 120
 defining start parameters 123
 importing template 119
 security 161
 template requirements 106
 TServerClientSap section 242
 call-timer-timeout option 242
 defer-tlib-events option 242
 waiting-for-newcall-timeout option 242
 TSingleStepTransfer 50
 TXRouteType 51
 type styles
 conventions 294
 italic 294
 monospace 294
 typographical styles 293, 294

U

Universal Routing 137
 UNIX
 installing server applications 127
 sockets 282
 starting server process 178
 stopping server process 179
 upgrading T-Server 136
 use-data-from
 configuration option 193
 use-implicit-access-numbers
 configuration option 197
 user data propagation 84
 user-data-limit
 configuration option 187

V

valid time units 239
 validation-scheme
 configuration option 243
 VDN 63
 verbose
 common log option 210
 version numbering, document 293

virtual
 switch, assigning 116
 switch, creating 116
 switching offices 115
 Virtual Routing Points 25
 VirtualRoutePoints section 247
 ExtensionDelimiter option 248
 premise T-Server option 247
 voice treatment ports 137

W

waiting-for-newcall-timeout
 configuration option 242
 warm standby 27, 40, 117, 123, 165
 figure 166
 T-Server configuration 170
 Windows
 installing server applications 128
 Services Manager 179
 starting server process 178
 stopping server process 179
 wizard 105
 configuration 131
 migration 132
 using manager 133

X

x-conn-debug-all
 common log option 224
 x-conn-debug-api
 common log option 223
 x-conn-debug-dns
 common log option 223
 x-conn-debug-open
 common log option 222
 x-conn-debug-security
 common log option 223
 x-conn-debug-select
 common log option 222
 x-conn-debug-timers
 common log option 222
 x-conn-debug-write
 common log option 222
 XML protocol 23
 XmlSap section 243
 source-encoding option 243
 target-encoding option 243
 validation-scheme option 243

